ANNOTATED BIBLIOGRAPHY ON HUMAN FACTORS IN SOFTWARE DEVELOPMENT

M. E. Atwood, et al

Science Applications, Incorporated
Englewood, Colorado

June 1979

ARI TECHNICAL REPORT
P-79-1

ANNOTATED BIBLIOGRAPHY ON HUMAN FACTORS IN SOFTWARE DEVELOPMENT

by

Michael E. Atwood and H. Rudy Ramsey
SCIENCE APPLICATIONS, INC.

Jean N. Hooper
ARMY RESEARCH INSTITUTE

Daniel A. Kullas
SCIENCE APPLICATIONS, INC.

June 1979

Contract DAHC 19-76-C-0040

Prepared for

U.S. ARMY RESEARCH INSTITUTE
for the BEHAVIORAL and SOCIAL SCIENCES
5001 Eisenhower Avenue
Alexandria, Virginia 22333

U. S. ARMY RESEARCH INSTITUTE FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency under the Jurisdiction of the Deputy Chief of Staff for Personnel

JOSEPH ZEIDNER
Technical Director

WILLIAM L. HAUSER
Colonel, US Army
Commander

Research accomplished under contract
to the Department of the Army

Science Applications, Incorporated

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: P-79-1	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): ANNOTATED BIBLIOGRAPHY ON HUMAN FACTORS IN SOFTWARE DEVELOPMENT	5. TYPE OF REPORT & PERIOD COVERED: Special Report: Bibliography 12 July 1978 - 11 Nov 1977
	6. PERFORMING ORG. REPORT NUMBER: SAI-79-102-DEN
7. AUTHOR(s): Michael E. Atwood, H. Rudy Ramsey, Jean N. Hooper (ARI), and Daniel A. Kullas	8. CONTRACT OR GRANT NUMBER(s): DAHC19-76-C-0040
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Science Applications, Inc. 7935 E. Prentice Avenue Englewood, CO 80111	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 2Q762725A778
11. CONTROLLING OFFICE NAME AND ADDRESS: U.S. Army Research Institute for the Behavioral and Social Sciences (PERI-OS) 5001 Eisenhower Avenue, Alexandria, VA 22333	12. REPORT DATE: June 1979
	13. NUMBER OF PAGES: 167
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): --	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: --

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

--

18. SUPPLEMENTARY NOTES

Monitored technically by Jean N. Hooper and Edgar M. Johnson, Human Factors Technical Area, ARI.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Human factors engineering	Maintenance	Programmers
Bibliographies	Management	
Computer programs	Reliability	
Computer Programming	Documentation	
Debugging (computers)	Programming languages	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

As part of a larger Army Research Institute effort to survey, synthesize, and evaluate the state of the art in the area of human factors as applied to software development, a fairly extensive literature survey was conducted. This resulting bibliography contains citations of 478 articles or reports pertaining to the behavioral aspects of software design, programming, coding, debugging, testing. evaluation, and maintenance. Most citations are accompanied by descriptive abstracts, and all are indexed by author, publication

(continued)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Item 20 (continued)

source, institutional affiliation, and subject. To help the user unfamiliar with the area, the bibliography contains brief, basic reference lists in the areas of software engineering, the psychology of software development, the Structured Programming Series, and the DoD software program. Coverage is exhaustive through 1977 with a few references in 1978.

FOREWORD

The Human Factors Technical Area is concerned with the human resource demands of increasingly complex battlefield systems used to acquire, transmit, process, disseminate, and utilize information. Research in this area is focused on human performance and interactions within command and control centers, as well as issues of system development. It is concerned with such areas as topographic products and procedures, tactical symbology, user-oriented systems, information management, staff operations and procedures, and sensor systems integration and utilization.

One area of special interest involves the development of computer software to support battlefield automated systems. Software development is a human process which is costly, inaccurate, and not well understood. The present report provides an extensive compilation of literature in the area of human factors aspects of software development; it was developed as a source document supporting human factors research in this area. This report is a part of a larger effort to develop a conceptualization of the programming process as a basis for identification and resolution of behavioral bottlenecks in software development. Efforts in this area are directed at improving accuracy and productivity in programming through the design of procedures, methods, and job aids to enhance programmer performance.

Research concerned with human factors in software development is conducted as an in-house effort augmented contractually by organizations selected as having unique capabilities and facilities. The present research was conducted in collaboration with personnel from Science Applications, Inc., under contract DAHC 19-76-C-0040. The effort is responsive to requirements of Army Project 2Q762725A778, and to general requirements expressed by members of the Integrated Software Research and Development Working Group (ISRAD).

JOSEPH ZEIDNER
Technical Director

ANNOTATED BIBLIOGRAPHY ON HUMAN FACTORS IN SOFTWARE DEVELOPMENT

CONTENTS

INTRODUCTION

The development of software for computer systems is a little-understood, labor-intensive activity. Computer software is developed by people, by systems analysts, software designers, and programmers, yet little is known about the individual and group processes whereby software is produced. The study of human factors in software production is a relatively new area to both psychologists and computer scientists; research investigating human performance in programming tasks has been published in widely scattered journals of psychology and computer science, books, and technical reports.

This bibliography was compiled to bring together in a single reference document literature from many sources related to behavioral aspects of the software development process. It is intended to provide an information source for psychologists, computer scientists, and other persons interested in the human factors of software production.

SCOPE OF THE BIBLIOGRAPHY

Software development, for the purpose of this bibliography, encompasses all phases of the process, including requirements analysis, software design, programming, coding, debugging testing, evaluation, and maintenance. The literature of concern deals with the processes of software development, including human performance in software design and programming tasks as well as with the human factors aspects of the tools and programming languages used by software developers. Literature dealing with the selection, training, and management of computer personnel is also included. However, papers dealing with software development, but lacking clear human factors results or discussion have been excluded. The bibliography provides comprehensive, although not exhaustive, coverage of the literature in the field through the end of 1977. Some articles have been included which were published in 1978.

The bibliography is supplemented by four reference lists which appear at the front of the document. In general, items on these reference lists are not cited in the bibliography. For the user of this bibliography who is new to human factors in software development, the first two lists provide a core of references to acquaint the user with software engineering and with related human factors research. The third reference list consists of the fifteen-volume Structured Programming Series. These reports, prepared by IBM for the Army and Air Force, provide background information and standards for the implementation of structured programming methodology using high order languages. Finally, the references to the DoD software program trace the development of the High Order Language Program and the Defense System Software Management Program.

ORGANIZATION OF THE BIBLIOGRAPHY

This bibliography consists of three parts: the four supplemental reference lists, the 478 literature citations and abstracts, and the literature indexes.

A typical bibliographic entry is shown in Figure 1, which identifies the components of the entry. The citation format follows the standards of the Publication Manual of the American Psychological Association (Second Edition) with one exception: both month and year of publication may be noted to make the document easier to locate. Where the same article appeared in multiple sources, the source thought to be more readily available (such as a journal article rather than company technical report) is generally given first.

The majority of the entires include an abstract. Where the abstract was unavailable but the document warranted inclusion, the citation appears without the abstract. Many different abstract styles and formats may be noted by the user of this bibliography; where available, author abstracts were used with no changes made, noted with "(A)" following the abstract. If the abstract appeared with the original document but was written by someone other than the author, it is noted by "(O)" after the abstract. Abstracts written by the bibliography authors are noted with their initials.

Four different classes of indexes are provided for the documents included in this bibliography: an author index, three document source indexes, an institutional affiliation index, and a subject index. The indexes direct the user to citations by document number, not by page number. The document source index consists of three groups, journals, proceedings, and agencies. Most documents within the bibliography are indexed by only one or two subject terms. Some index terms are admittedly broad, but the user is referred to additional index categories which are related.

① 35 ② STATISTICS ON USE OF INTERACTIVE DEBUGGING FACILITIES

③ BOIES, S.J., & SPIEGEL, M.F. A BEHAVIORAL ANALYSIS OF PROGRAMMING: ON THE USE OF INTERACTIVE DEBUGGING FACILITIES (TECHNICAL REPORT RC-4472). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, AUGUST 1973. ④ (NTIS NO. AD 772127)

DESCRIPTION:

⑤ MEASUREMENT AND ANALYSIS OF USER DEBUGGING BEHAVIOR ON PRESENT INTERACTIVE SYSTEMS CAN LEAD TO AN UNDERSTANDING OF HOW FUTURE INTERACTIVE SYSTEMS SHOULD BE IMPLEMENTED. THIS STUDY PRESENTS BOTH AN ANALYSIS OF THE ON-LINE USAGE OF THE TSS/360 PROGRAM CONTROL SYSTEM (PCS) AS WELL AS THE RESULTS OF A QUESTIONNAIRE DESIGNED TO PROBE THE PCS KNOWLEDGE OF THE POPULATION OF USERS OF THAT SYSTEM. THE QUESTIONNAIRE RESPONSE POPULATION TENDED TO BE DIVIDED INTO THREE COMPETENCE GROUPS, THE HIGHEST OF WHICH CONSISTED MOSTLY OF SYSTEM SUPPORT PERSONNEL AND ASSEMBLER LANGUAGE PROGRAMMERS. THE ON-LINE USAGE ANALYSIS SHOWED THAT PCS COMMANDS ACCOUNTED FOR 7.2 PERCENT OF ALL COMMANDS ENTERED BY PROGRAMMERS. PCS WAS USED RARELY TO MODIFY THE LOGIC OF A PROGRAM, BUT WAS USED AS A PROGRAM VARIABLE INPUT/OUTPUT SYSTEM. IT WAS CONCLUDED THAT THE UNDERSTANDING OF THE INTERACTIVE DEBUGGING PROCESS MUST COME FROM THE PERSPECTIVE OF THE ENTIRE SYSTEM, AS UP TO 50 PERCENT OF THE DEBUGGING EFFORT IS SPENT ON TASK AND DATA MANAGEMENT. FIVE CRITICAL FEATURES OF AN EFFECTIVE INTERACTIVE PROGRAMMING SYSTEM ARE OUTLINED. (A) ⑥

⑦ 25P, 12R. ⑧

1. Document number
2. Subject heading
3. Reference citation
4. NTIS accession number
5. Abstract
6. Abstract source
7. Number of pages in article
8. Number of references in article

Figure 1. Sample bibliographic entry

BASIC REFERENCES: SOFTWARE ENGINEERING

Aron, J. D. The program development process: Part I. The individual programmer. Reading, MA: Addison-Wesley, 1974.

Boehm, B. W. Software and its impact: A quantitative assessment. Datamation, 1973, 19, 49-59.

Boehm, B. W. Software engineering. IEEE Transactions on Computers, 1976, C-25, 1226-1241.

Brooks, F. The mythical man-month: Essays on software engineering. Reading, MA: Addison-Wesley, 1975.

Buxton, J. M., Naur, P., & Randell, B. Software engineering: Concepts and techniques. Proceedings of the NATO Conferences, Petrocelli/Charter, New York, 1976.

Cougar, J. D., & Knapp, R. W. System analysis techniques. New York: John Wiley & Sons, 1974.

Dahl, O. J., Dijkstra, E. W., & Hoare, C. A. R. Structured programming. New York: Academic Press, 1974.

Daly, E. B. Management of software development. IEEE Transactions on Software Engineering, 1977, SE-3, 230-242.

Dijkstra, E. W. A discipline of programming. Englewood Cliffs, NH: Prentice-Hall, Inc., 1976.

Freeman, P. & Wasserman, A. I. (Eds.) Tutorial on software design techniques (2nd Ed.). Long Beach, CA: IEEE Computer Society, 1977.

Gilb, T. Software metrics. Cambridge, MA: Winthrop Publishers, 1977.

Halstead, M. H. Elements of software science. New York: Elsevier North-Holland, 1977.

Holdt, R. C. Structure of computer programs: A survey. Proceedings of the IEEE, 1975, 63, 879-893.

Horowitz, E. (Ed.) Practical strategies for developing large software systems. Reading, MA: Addison-Wesley, 1975.

Jackson, M. A. Principles of program design. New York: Academic Press, 1975.

Kernighan, B. W., & Plauger, P. J. The elements of programming style. New York: McGraw-Hill, 1974 (Also reported more briefly in Programming style: Examples and counterexamples. Computing Surveys, 1974, 6, 303-319).

Kernighan, B. W., & Plauger, P. J. Software tools. Reading, MA: Addison-Wesley, 1976.

Kosy, D. W. Air Force command and control information processing in 1980s: Trends in software technology (Technical Report No. R-1012-PR). Santa Monica, CA: Rand Corporation, June 1974.

Ledgard, H. F. Programming proverbs. Rochelle Park, NJ: Hayden, 1975.

Ledgard, H. F. Programming proverbs for FORTRAN programmers. Rochelle Park, NJ: Hayden, 1975.

Mills, H. D. Software engineering. Science, 1977, 195, 1199-1204.

Myers, G. J. Software reliability: Principles and practices. New York: John Wiley & Sons, 1976.

Myers, W. The need for software engineering. Computer, 1978, 11 (2), 12-26.

Sammet, J. E. Programming languages: History and fundamentals. Englewood Cliffs, NJ: Prentice-Hall, 1969.

Sammet, J. E. Roster of programming languages for 1976-77. SIGPLAN Notices, 1978, 13 (11), 56-85. (Separate rosters were published for each of the years 1967 through 1972 and for 1974-75.)

Van Tassel, D. Program style, design, efficiency, debugging, and testing. Englewood Cliffs, NJ: Prentice-Hall, 1974.

Wegner, P. Programming languages: The first 25 years. IEEE Transactions on Computers, 1976, C-25, 1207-1225.

Weinwurm, G. F. (Ed.) On the management of computer programming. New York, NY: Auerbach, 1970.

Wirth, N. Systematic programming: An introduction. Englewood Cliffs, NJ. Prentice-Hall, 1973.

Wirth, N. On the composition of well structured programs. Computing Surveys, 1974, 6, 247-259.

Wolverton, R. W. The cost of developing large-scale software. IEEE Transactions on Computers, 1974, C-23, 615-636.

Yourdon, E., & Constantine, L. L. Structured design. New York: Yourdon, Inc., 1975.

Zelkowitz, M. V. Perspectives on software engineering. ACM Computing Surveys, 1978, 10, 197-216.

BASIC REFERENCES IN
PSYCHOLOGY OF SOFTWARE ENGINEERING

Berger, R. M. Computer programmer job analysis reference text. Montvale, NJ: American Federation of Information Processing Societies, Inc., 1974.

Boies, S.J. User behavior on an interactive computer system. IBM Systems Journal, 1974, 13, 2-18.

Boies, S.J., & Gould, J.D. Syntactic errors in computer programming. Human Factors, 1974, 16, 253-257.

Brooks, R. A model of human cognitive behavior in writing code for computer programs (2 Vols.). Pittsburgh, PA: Carnegie-Mellon University, Department of Computer Science, May 1975 (Also Reported briefly in Proceedings of the Fourth International Joint Conference of Artificial Intelligence, 1975). (NTIS Nos. AD A013582, AD A012918).

Cooke, J. E., & Bunt, R. R. Human error in programming: The need to study the individual programmer. Infor, 1975, 13, 296-307 (Also Technical Report No. 75-3, Department of Computational Sciences, University of Saskatoon, Canada, 1975).

Endres, A. An analysis of errors and their causes in system programs. Proceedings, International Conference on Reliable Software, 21-23 April 1975, Los Angeles, CA. SIGPLAN Notices, 1975, 10, 131-142 (Also IEEE Transactions of Software Engineering, 1975, SE-1, 140-149).

Gannon, J. D. An experiment for the evaluation of language features. International Journal of Man-Machine Studies, 1976, 8, 61-73.

Gannon, J. D. & Horning, J. J. Language design for programming reliability. IEEE Transactions on Software Engineering, 1975, SE-1, 179-191.

Gould, J. D. & Drongowski, P. An exploratory study of computer program debugging. Human Factors, 1974, 16, 258-277.

Green, T. R. G., Sime, M. E. & Fitter, M. Behavioral experiments on programming languages: Some methodological considerations. MRC Memo No. 66. Sheffield, England: Sheffield University, Department of Psychology, 1975.

Halpern, M. Foundations of the case for natural-language programming. IEEE Spectrum, 1967, 4, 140-149.

Hoc, J. M. The role of mental representation in learning a programming language. International Journal of Man-Machine Studies, 1977, 9, 87-105.

Miller, L. A. Programming by non-programmers. International Journal of Man-Machine Studies, 1974, 6, 237-260 (Also Research Report RC-4280, IBM Watson Research Center, Yorktown Heights, New York, 1973).

Miller, L. A. Naive programmer problems with specification of transfer-of-control. AFIPS Conference Proceedings, 1975, 44, 657-663.

Miller, L. A. & Becker, C. A. Programming in natural English (Technical Report RC-5137). Yorktown Heights, NY: IBM Watson Research Center, November 1974. (NTIS No. AD A003923)

Parsons, H. M. The scope of human factors in computer-based data processing systems. Human Factors, 1970, 12, 165-175.

Reisner, P. Use of psychological experimentation as an aid to development of a query language. IEEE Transactions on Software Engineering, 1977, SE-3, 218-229.

Sackman, H. Man-computer problem solving. Princeton, NJ: Auerbach, 1970.

Scott, R. F. & Simmons, D. B. Predicting programming group productivity: A communications model. IEEE Transactions on Software Engineering, 1975, SE-1, 411-414.

Shneiderman, B. Exploratory experiments in programmer behavior. International Journal of Computer and Information Sciences, 1976, 5, 123-143.

Shneiderman, B., Mayer, R., McKay, D., & Heller, P. Experimental investigations of the utility of flowcharts in programming. Communications of the ACM, 1977, 20, 373-381.

Sime, M. E., Green, T. R. G. & Guest, D. J. Psychological evaluation of two conditional constructions used in computer languages. International Journal of Man-Machine Studies, 1973, 5, 105-113.

Sime, M. E., Green, T. R. G. & Guest, D. J. Scope marking in computer conditionals: A psychological evaluation. International Journal of Man-Machine Studies, 1977, 9, 107-118.

Taylor, I. C. Computer language and natural language taught comparatively. Computers and People, 1977, 26, 7-11.

Weinberg, G. M. The Psychology of Computer Programming. New York: Van Nostrand Reinhold, 1971.

Weinberg, G. M. & Schulman, E. L. Goals and performance in computer programming. Human Factors, 1974, 16, 70-77.

Weissman, L. M. A methodology for studying the psychological complexity of computer programs (Technical Report 37). Toronto, Ontario, Canada: University of Toronto, Computer Systems Research Group, August 1974.

Youngs, E. A. Human errors in programming. International Journal of Man-Machine Studies, 1974, 6, 351-376.

STRUCTURED PROGRAMMING SERIES

Kessler, M. M., & Tinanoff, N. Structured programming series (Vol I): Programming language standards (Report No. RADC-TR-74-300, Vol. I). Griffiss Air Force Base, NY: Rome Air Development Center, March 1975.

Tinanoff, N. Structured programming series (Vol. II): Precompiler specifications (Report No. RADC-TR-74-300, Vol. II). Griffiss Air Force Base, NY: Rome Air Development Center, May, 1975.

Campbell, D. G. & Kessler, M. M. Structured programming series (Vol. III): ANS COBOL precompiler program documentation (Report No. RADC-TR-74-300, Vol. III). Griffiss Air Force Base, NY: Rome Air Development Center, March, 1975.

Trimble, J. T. Structured programming series (Vol. IV): Data structuring study (Report No. RADC-TR-74-300, Vol. IV). Griffiss Air Force Base, NY: Rome Air Development Center, April, 1975.

Luppino, F. M. & Smith R. L. Structured programming series (Vol. V): Programming Support Library (PSL) program specifications (Report No. RADC-TR-74-300, Vol. VI). Griffiss Air Force Base, NY: Rome Air Development Center, July, 1974.

Tinanoff, N. & Luppino, F. M. Structured programming series (Vol. VI): Programming Support Library (PSL) program specifications (Report No. RADC-TR-74-300, Vol. VI). Griffiss Air Force BAse, NY: Rome Air Development Center, November, 1974.

Ortega, L. H. Structured programming series (Vol. VII): Documentation standards (Report No. RADC-TR-74-300, Vol. VII). Griffiss Air Force Base, NY: Rome Air Development Center, September, 1974.

Kraly, T. M., Naughton, J. J., Smith, R. L., & Tinanoff, N. Structured programming series (Vol. VIII): Program design study (Report No. RADC-TR-74-300, Vol. VIII). Griffiss Air Force Base, NY: Rome Air Development Center, May, 1975.

Smith, R. L. Structured programming series (Vol. IX): Management data collection and reporting (Report No. RADC-TR-74-300, Vol. IX). Griffiss Air Force Base, NY: Rome Air Developmental Center, October, 1974.

Barry, B. J. & Naughton, J. J. Structured programming series (Vol. X): Chief programmer team operations description (Report No. RADC-TR-74-300, Vol. X). Griffiss Air Force Base, NY: Rome Air Development Center, January, 1975.

Smith, R. L. Structured programming series (Vol. XI): Estimating software project resource requirements (Report No. RADC-TR-74-300, Vol XI). Griffiss Air Force Base, NY: Rome Air Development Center, January, 1975.

Kessler, M. M. Structured programming series (Vol. XII): Training materials (Report No. RADC-TR-74-300, Vol. XII). Griffiss Air Force Base, NY: Rome Air Development Center, July, 1975.

Naughton, J. J. & Smith, R. L. Structured programming series (Vol. XIII): Final report (Report No. RADC-TR-74-300, Vol. XIII). Griffiss Air Force Base, NY: Rome Air Development Center, July, 1975.

Kessler, M. M. & Kister, W. E. Structured programming series (Vol. XIV): Software tool impact (Report No. RADC-TR-74-300, Vol. XIV). Griffiss Air Force Base, NY: Rome Air Development Center, May, 1975.

Smith, R. L. Structured programming series (Vol. XV): Validation and verification study (Report No. RADC-TR-74-300, Vol. XV). Griffiss Air Force Base, NY: Rome Air Development Center, May, 1975.

BASIC REFERENCES TO THE DOD SOFTWARE PROGRAM

Defense System Software Management Program. Washington, DC: Office of the Assistant Secretary of Defense for Installations and Logistics March 1976.

Defense System Software Research and Development Technology Plan. In preparation.

Management of Computer Resources in Major Defense Systems. Department of Defense Systems Directive No. 5000.29, April 26, 1976.

Interim List of DOD Approved High Order Programming Languages. Department of Defense Instruction No. 5000.31, November 24, 1976.

HIGH ORDER LANGUAGE PROGRAM

Department of Defense Requirements for High Order Computer Programming Languages, "Ironman," January 1977. Point of Contact: LTC William A. Whitaker, Defense Advanced Research Projects Agency, 1400 Wilson Blvd., Arlington, VA 22209.

Department of Defense Requirements for High Order Computer Programming Languages, "Tinman," March 1976.

Fisher, D. A. A common programming language for the Department of Defense--background and technical requirements. IDA paper P-1191, Institute for Defense Analysis, June 1976. (AD-A028 297)

SUPPORTING ANALYSES

Kossiakoff, A., et al. DOD Weapon Systems Software Management Study. The Johns Hopkins University Applied Physics Laboratory, Report No. SR 75-3.

Kossiakoff, A., et al. DOD Weapon Systems Software Management Study, Appendix A, Findings and Recommendations of Previous Studies. The Johns Hopkins University Applied Physics Laboratory, Report No. SR 73-3A, June 1975.

Asch, A., et al. DOD Weapon Systems Software Acquisition and Management Study, Vol I, Findings and Recommendations. MITRE Corporation Report No. MTR-6908 Vol I, May 1975.

Asch, A., et al. DOD Weapon Systems Software Acquisition and Management Study, Vol II, Supporting Material. MITRE Corporation Report No. MTR-6908 Vol II, June 1975.

BASELINE STUDIES

Gates, H. P., Jr., Gourary, B. S., & Deitchman, S. J. Electronics-X: A Study of Military Electronics with Particular Reference to Cost and Reliability. Volume 2: Complete Report. Arlington, VA: Institute for Defense Analyses, R-195, January 1974.

Reich, E. T. Tactical Computer Software Acquisition and Maintenance Staff Study. Office of the Assistant Secretary of Defense, Washington, DC. October 1973.

Report of the Army Scientific Advisory Panel Ad Hoc Committee for Army Tactical Data System Software Development, Army Scientific Advisory Panel, Washington, DC, October 1974.

Boehm, B. W. & Haile, A. C. Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980s (CCIP-85). Executive Summary (Revised Edition), USAF Space and Missile Systems Organization, Los Angeles, CA, SAMSO TR72-122, February 1972.

Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980s (CCIP-85). Volume I. Highlights, USAF Space and Missile Systems Organization, Los Angeles, CA, SAMSO TR 72-141, April 1972.

The CCIP-85 study also produced ten supporting volumes:

Volume II	Command and Control Requirements: Overview Annex A: Strategic Requirements Annex B: Air Defense Requirements Annex C: Tactical Requirements
Volume III	Command and Control Requirements: Intelligence
Volume IV	Technology Trends: Software
Volume V	Technology Trends: Hardware
Volume VI	Technology Trends: Sensors
Volume VII	Technology Trends: Integrated Design
Volume VIII	Interservice Coordination Trends
Volume IX	Analysis
Volume X	Current Research and Development
Volume XI	Roadmaps

Project Pacer Flash--Volume I. Executive Study and Final Report, Air Force Logistics Command, Wright-Patterson AFB, OH, September 1973.

Fisher, D. A., Jr. Automatic Data Processing Costs in the Defense Department. Arlington, VA: Institute for Defense Analysis, IDA-P-1046, October 1974.

A Report on Air Force Logistics Command Operation Flight Program Support. Santa Monica, CA: System Development Corp., TM-5439/000/00 and TM-5439/001/00, December 1974.

Proceedings of the Aeronautical Systems Software Workshop (Draft). Air Force Systems Command, Washington, D.C., April 1974.

Proceedings of a Symposium on the High Cost of Software. September 17-19, 1973. Menlo Park, CA: Stanford Research Institute, SRI Project 3272, September 1973. (AD 777121)

Summary Notes of a Government/Industry Sizing and Costing Workshop, Electronic Systems Division, Hanscom AFB, MA, February 1975.

1 PROGRAMMING

ABRAMS, P.S. PROGRAM WRITING, REWRITING, AND STYLE. IN P. GJERLOV, H.J. HELMS, & J. NIELSEN (EDS.), APL CONGRESS 73: PROCEEDINGS OF THE APL CONGRESS 73. NEW YORK: AMERICAN ELSEVIER PUBLISHING CO., 1973, 1-8.

DESCRIPTION:

THIS PAPER DISCUSSES THE PROCESSES AND PROBLEMS OF WRITING AND ESPECIALLY OF REWRITING PROGRAMS. REWRITING IS DEFINED AS DERIVING A NEW PROGRAM USING THE SAME ALGORITHM AS THE ORIGINAL. THERE ARE SEVERAL REASONS FOR REWRITING A PROGRAM -- TO ATTEMPT TO OPTIMIZE IT, TO SIMPLIFY IT, OR TO PUT IT IN A FORM THAT IS EASIER TO UNDERSTAND. THIS PAPER TRACES THE EVOLUTION OF A PROGRAM THROUGH SEVERAL REWRITING STEPS. FOR THE MOST PART, THESE REFINEMENTS ARE BASED ON A FORMAL ANALYSIS OF THE PROGRAM TEXT, ALTHOUGH KNOWLEDGE OF THE SPECIFIC PROBLEM AND THE BEHAVIOR OF THE ALGORITHM ARE ALSO USED AT SEVERAL POINTS. (PEA)

8P, 4R.

2 TIME-SHARING VERSUS BATCH PROCESSING

ADAMS, J., & COHEN, L. TIME-SHARING VS. INSTANT BATCH PROCESSING: AN EXPERIMENT IN PROGRAMMER TRAINING. COMPUTERS AND AUTOMATION, MARCH 1969, 18(3), 30-34.

DESCRIPTION:

EIGHT STUDENT PROGRAMMERS STUDIED FORTRAN FOR TWO WEEKS, ONE WEEK USING A CRT-BASED TIME-SHARING SYSTEM AND ONE WEEK USING "INSTANT BATCH", IN A COUNTERBALANCED REPEATED-MEASURES DESIGN. BECAUSE OF LARGE INDIVIDUAL DIFFERENCES, NO SIGNIFICANT DIFFERENCES IN OBJECTIVE MEASURES WERE DETECTED. IN QUESTIONNAIRE RESULTS, HOWEVER, THE STUDENTS INDICATED A STRONG PREFERENCE FOR THE BATCH MODE. GIVEN A CHOICE, ALL STUDENTS USED THE BATCH SYSTEM FOR THE REMAINDER OF THE SUMMER. (HRR)

5P, 5R.

3 DEBUGGING

AKIYAMA, F. AN EXAMPLE OF SOFTWARE SYSTEM DEBUGGING. PROCEEDINGS OF THE IFIP CONGRESS, 1971, 353-359.

DESCRIPTION:

IN THE SOFTWARE DEVELOPMENT PROCESS, THE PROGRAMMING AND DEBUGGING STAGE OF A PROGRAM COMES LAST, I.E., IMMEDIATELY BEFORE DELIVERY, AND THE NUMBER OF BUGS TO BE FOUND THEN IS QUITE UNPREDICTABLE BEFOREHAND. THIS IS THE REASON WHY THE PLANNED PROCESS IS DISTURBED AND THE DELIVERY TIME DELAYED IN SO MANY CASES. IN ORDER TO AVOID THE DELAY IN SCHEDULE, IT IS NECESSARY TO PREDICT QUANTITATIVELY THE NUMBER OF BUGS AND, ALSO, THE MANHOURS NECESSARY FOR DEBUGGING. THE NUMBER OF BUGS MAY BE ESTIMATED BASED ON THE NATURE OF A PROGRAM. HENCE, BY CAREFULLY STUDYING A GENERAL FLOW CHART DURING THE DETAILED DESIGNING OF SOFTWARE, WE CAN PREDICT THE NUMBER OF BUGS IN THE SYSTEM, AND DECIDE ON THE NUMBER OF CHECK POINTS TO BE MADE DURING THE PROCESS AND, CONSEQUENTLY, THE MANHOURS NECESSARY. (A)

7P, 7R.

4 PROGRAMMING LANGUAGES

ALEXANDER, W.G. HOW A PROGRAMMING LANGUAGE IS USED (TECHNICAL REPORT CSRG-10). TORONTO, ONTARIO, CANADA: UNIVERSITY OF TORONTO, COMPUTER SYSTEMS RESEARCH GROUP, FEBRUARY 1972.

DESCRIPTION:

AN EMPIRICAL STUDY OF PROGRAMS WRITTEN IN XPL WAS CARRIED OUT WITH THE AIM OF DETERMINING PROPERTIES OF BOTH THE LANGUAGE AND THE OBJECT MACHINE, THE S/360. THE INFORMATION GATHERED BY EXAMINING A SET OF TYPICAL XPL PROGRAMS WAS USED TO SUGGEST IMPROVEMENTS TO THE DESIGN OF XPL, TO THE COMPILER FOR XPL, AND TO THE S/360. IN ADDITION, A PROFILE GENERATOR FOR XPL PROGRAMS WAS DEVELOPED AND ILLUSTRATED. A POWERFUL PROGRAMMING TOOL, IT ENABLES AN XPL PROGRAMMER TO CLEARLY SEE WHERE EXECUTION TIME IS SPENT IN HIS PROGRAM. (A)

119P, 11R.

5 PROGRAMMING

ALSPAUGH, C.A. IDENTIFICATION OF SOME COMPONENTS OF COMPUTER PROGRAMMING APTITUDE. JOURNAL FOR RESEARCH IN MATHEMATICS EDUCATION, 1972, 3, 89-98.

DESCRIPTION:

AN EXPERIMENT WAS CONDUCTED TO DETERMINE THE EXTENT THAT THE THURSTONE TEMPERAMENT SCHEDULE, THE IBM PROGRAMMER APTITUDE TEST, THE WATSON-GLASER CRITICAL THINKING APPRAISAL, THE SCAT QUANTITATIVE AND VERBAL SUBTESTS, AND MATHEMATICAL BACKGROUND CORRELATE WITH A MEASURE OF PROGRAMMING PROFICIENCY (TEST SCORES) IN AN INTRODUCTORY COMPUTER SCIENCE COURSE USING BASIC ASSEMBLY LANGUAGE (BAL) AND FORTRAN. NO SIGNIFICANT DIFFERENCES WERE OBSERVED BETWEEN BAL AND FORTRAN ON EITHER THE PROFICIENCY MEASURE OR THEIR CORRELATIONS WITH THE INDEPENDENT VARIABLES STUDIED. THE MATHEMATICAL BACKGROUND OF A STUDENT WAS THE MAJOR PREDICTOR OF PERFORMANCE. THE MORE SUCCESSFUL STUDENTS ALSO TEND TO HAVE LOW LEVELS OF "IMPULSIVENESS" AND "SOCIABILITY" AND A RELATIVELY HIGH LEVEL OF "REFLECTIVENESS" AS MEASURED BY THE THURSTONE TEMPERAMENT SCHEDULE. (MEA)

10P, 9R.

6 SOCIAL IMPACTS OF COMPUTING

ANDERSON, R.E. VALUE ORIENTATION OF COMPUTER SCIENCE STUDENTS. COMMUNICATIONS OF THE ACM, 1978, 21, 219-225.

DESCRIPTION:

TECHNOLOGICAL AND NON-TECHNOLOGICAL VALUE ORIENTATIONS ARE INVESTIGATED WITH SPECIAL ATTENTION TO THE COMPLEXITY OF VALUE STRUCTURES. COMPUTER SCIENCE STUDENTS, WHO ARE CLOSELY ASSOCIATED WITH TECHNOLOGY, CONTRAST WITH SOCIAL SCIENCE STUDENTS, WHO ARE OFTEN TECHNOLOGICALLY ALOOF. THIS IS CONFIRMED BY THE VALUE RATINGS OF 313 STUDENTS AT THE UNIVERSITY OF MINNESOTA IN 1972. COMPUTER SCIENCE MAJORS WERE FOUND TO HAVE A MORE COMPLEX VALUE STRUCTURE THAN SOCIAL SCIENCE MAJORS. (A)

7P, 29R.

7 PROGRAMMING

ARBLASTER, A. SOME MEASURES OF INFORMATION ABOUT PROGRAM STATES. IN E. MORLET & D. RIBBENS (EDS.), INTERNATIONAL COMPUTING SYMPOSIUM. NORTH HOLLAND PUBLISHING COMPANY, 1977, 183-190.

DESCRIPTION:

SOME MEASURES OF INFORMATION ABOUT THE SEMANTIC STATE OF PROGRAMS ARE CONSIDERED. A METHOD FOR DEFINING MEASURES FOR A VARIETY OF DATA TYPES IS FIRST DISCUSSED, THEN SOME MEASURES OF THE RELATIONSHIP BETWEEN TESTS AND ACTIONS IN CONDITIONALS ARE EXAMINED. (A)

9P, 29R.

8 STRUCTURED PROGRAMMING

ATKINSON, G. THE NON-DESIRABILITY OF STRUCTURED PROGRAMMING IN USER LANGUAGES. SIGPLAN NOTICES, JULY 1977, 12(7), 43-50.

DESCRIPTION:

THE APPLICABILITY OF STRUCTURED PROGRAMMING CONCEPTS IN THE DESIGN OF SPECIAL-PURPOSE LANGUAGES FOR THE NONPROGRAMMING "CASUAL USER" IS EXAMINED. DESIGN CRITERIA FOR A CASUAL USER LANGUAGE ARE PRESENTED ALONG WITH A MODUS OPERANDI (THE 4:55 FRIDAY APPROACH) FOR DEVELOPING "NATURAL NOTATIONS" WHICH ARE EASILY LEARNED AND USED. AN EXAMPLE OF A CASUAL USER LANGUAGE IS PRESENTED ALONG WITH EVIDENCE THAT, FOR THE CASUAL USER, THE ABILITY TO WRITE OR EVEN RECOGNIZE A WELL-STRUCTURED PROGRAM IS AN UNNECESSARY BURDEN. (A)

3P, R.

9 PROGRAM DEBUGGING

ATWOOD, M.E., & RAMSEY, H.R. COGNITIVE STRUCTURES IN THE COMPREHENSION AND MEMORY OF COMPUTER PROGRAMS: AN INVESTIGATION OF COMPUTER PROGRAM DEBUGGING (TECHNICAL REPORT TR-78-A21). ALEXANDRIA, VIRGINIA: U.S. ARMY RESEARCH INSTITUTE FOR THE BEHAVIORAL AND SOCIAL SCIENCES, AUGUST 1978.

DESCRIPTION:

A THEORETICAL FRAMEWORK, BASED UPON RECENT STUDIES IN COGNITIVE PSYCHOLOGY ON MEMORY FOR TEXT, WAS DEVELOPED TO EXPLAIN CERTAIN ASPECTS OF HUMAN BEHAVIOR DURING COMPUTER PROGRAM COMPREHENSION AND DEBUGGING. A CENTRAL CONCEPT OF THIS FRAMEWORK IS THAT THE INFORMATION CONTAINED IN A PROGRAM IS REPRESENTED IN A PROGRAMMER'S MEMORY AS A CONNECTED, PARTIALLY ORDERED LIST (HIERARCHY) OF "PROPOSITIONS" (UNITS OF INFORMATION WITH PROPERTIES SIMILAR TO THOSE OBSERVED IN RESEARCH ON TEXT MEMORY). AN EXPERIMENT WAS PERFORMED TO TEST THE HYPOTHESIS THAT THE DIFFICULTY IN FINDING A PROGRAM BUG IS A FUNCTION OF THE BUG'S LOCATION IN THIS HIERARCHY. THIS EXPERIMENT COMPARED THE EFFECTS OF BUG LOCATION, BUG TYPE (ARRAY, ITERATION, ASSIGNMENT) AND SPECIFIC PROGRAM. EACH OF 48 SUBJECTS DEBUGGED TWO SEPARATE PROGRAMS, WITH ONE TYPE OF BUG AT TWO DIFFERENT HIERARCHICAL LEVELS IN EACH PROGRAM.

A PRELIMINARY ANALYSIS SUGGESTED THAT ALL THREE FACTORS -- PROGRAM, BUG TYPE, AND BUG LOCATION -- SIGNIFICANTLY AFFECTED THE TIME REQUIRED TO LOCATE PROGRAM BUGS. DETAILED ANALYSES, HOWEVER, SUGGESTED THE PROGRAM AND BUG TYPE VARIABLES COULD BE EXPLAINED IN TERMS OF THE BUG LOCATION VARIABLE AND THAT A BUG'S LOCATION IN A PROGRAM'S UNDERLYING PROPOSITIONAL HIERARCHY IS A PRINCIPAL FACTOR AFFECTING PERFORMANCE IN A COMPREHENSION AND DEBUGGING TASK. (A)

10 JOB SATISFACTION

AWAD, E.M. JOB SATISFACTION AS A PREDICTOR OF TENURE. COMPUTER PERSONNEL, 1977, 7(1-2), 7-10.

11 ERRORS

BAILEY, R.W., DEMERS, S.T., & LEBOWITZ, A.I. HUMAN RELIABILITY IN COMPUTER-BASED BUSINESS INFORMATION SYSTEMS. IEEE TRANSACTIONS ON RELIABILITY, AUGUST 1973, R-22(3), 140-148.

12 PROGRAMMING GROUPS

BAKER, F.T. CHIEF PROGRAMMER TEAM MANAGEMENT OF PRODUCTION PROGRAMMING. IBM SYSTEMS JOURNAL, 1972, 11, 56-73.

13 STRUCTURED PROGRAMMING

BAKER, F.T. SYSTEM QUALITY THROUGH STRUCTURED PROGRAMMING. AFIPS CONFERENCE PROCEEDINGS, 1972, 41, 339-343.

DESCRIPTION:

EXPERIENCE IN DEVELOPMENT AND MAINTENANCE OF LARGE COMPUTER-BASED SYSTEMS FOR GOVERNMENT AND INDUSTRY HAS LED THE IBM FEDERAL SYSTEMS DIVISION TO THE FORMULATION OF A NEW APPROACH TO PRODUCTION PROGRAMMING. THIS APPROACH, WHICH COUPLES A NEW KIND OF PROGRAMMING ORGANIZATION (A CHIEF PROGRAMMER TEAM) WITH FORMAL TOOLS FOR USING STRUCTURED PROGRAMMING IN SYSTEM DEVELOPMENT, WAS RECENTLY APPLIED ON A CONTRACT WITH THE NEW YORK TIMES FOR AN ONLINE INFORMATION SYSTEM. COMPARED TO EXPERIENCE ON SIMILAR CONTRACTS IN THE PAST, THE APPROACH RESULTED IN INCREASED PROGRAMMER PRODUCTIVITY COUPLED WITH IMPROVED QUALITY. FOLLOWING A BRIEF DESCRIPTION OF THE SYSTEM AND A REVIEW OF THE APPROACH, THIS PAPER DISCUSSES THE QUALITY OF THE SYSTEM AS OBSERVED DURING A THOROUGH ACCEPTANCE TEST AND IN THE INITIAL PERIOD OF OPERATION FOLLOWING ITS DELIVERY. (A)

SP, TR.

14 STRUCTURED PROGRAMMING

BAKER, F.T. STRUCTURED PROGRAMMING IN A PRODUCTION PROGRAMMING ENVIRONMENT. IN PROCEEDINGS, 1975 INTERNATIONAL CONFERENCE ON RELIABLE SOFTWARE. SIGPLAN NOTICES, JUNE 1975, 10(6), 172-185 (ALSO IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 241-252).

DESCRIPTION:

THIS PAPER DISCUSSES HOW STRUCTURED PROGRAMMING METHODOLOGY HAS BEEN INTRODUCED INTO A LARGE PRODUCTION PROGRAMMING ORGANIZATION USING AN INTEGRATED BUT FLEXIBLE APPROACH. IT NEXT ANALYZES THE ADVANTAGES AND DISADVANTAGES OF EACH COMPONENT OF THE METHODOLOGY AND PRESENTS SOME QUANTITATIVE RESULTS ON ITS USE. IT CONCLUDES WITH RECOMMENDATIONS BASED ON THIS GENERALLY SUCCESSFUL EXPERIENCE, WHICH COULD BE USEFUL TO OTHER ORGANIZATIONS INTERESTED IN IMPROVING RELIABILITY AND PRODUCTIVITY. (A) 14P, 18R.

15 SYSTEM REQUIREMENTS DEFINITION

BALZER, R., GOLDMAN, N., & WILE, D. INFORMALITY IN PROGRAM SPECIFICATIONS (TECHNICAL REPORT NO. ISI/RR-77-59). MARINA DEL REY, CALIFORNIA: UNIVERSITY OF SOUTHERN CALIFORNIA, INFORMATION SCIENCES INSTITUTE, APRIL 1977. (NTIS NO. AD A041669)

DESCRIPTION:

THIS WORK IS CONCERNED PRIMARILY WITH (1) THE PROCEDURE BY WHICH PROCESS-ORIENTED SPECIFICATIONS ARE OBTAINED FROM GOAL-ORIENTED REQUIREMENT SPECIFICATIONS AND (2) COMPUTER BASED TOOLS FOR THEIR CONSTRUCTION. IT FIRST DETERMINES SOME ATTRIBUTES OF A SUITABLE PROCESS-ORIENTED SPECIFICATION LANGUAGE, THEN EXAMINES THE REASONS WHY SPECIFICATIONS WOULD STILL BE DIFFICULT TO WRITE IN SUCH A LANGUAGE. THE KEY TO OVERCOMING THESE DIFFICULTIES SEEMS TO BE THE CAREFUL INTRODUCTION OF INFORMALITY (I.E., PARTIAL, RATHER THAN COMPLETE, DESCRIPTIONS) AND THE USE OF A COMPUTER-BASED TOOL THAT USES CONTEXT EXTENSIVELY TO COMPLETE THESE DESCRIPTIONS DURING THE PROCESS OF CONSTRUCTING A WELL-FORMED SPECIFICATION. SOME RESULTS OBTAINED BY A RUNNING PROTOTYPE OF SUCH A COMPUTER-BASED TOOL ON A FEW INFORMAL EXAMPLE SPECIFICATIONS ARE PRESENTED AND, FINALLY, SOME OF THE TECHNIQUES USED BY THIS PROTOTYPE SYSTEM ARE DISCUSSED. (A)

16 COMPUTER-ASSISTED INSTRUCTION

BARR, A., BEARD, M., & ATKINSON, R.C. A RATIONALE AND DESCRIPTION OF A CAI PROGRAM TO TEACH THE BASIC PROGRAMMING LANGUAGE. INSTRUCTIONAL SCIENCE, 1975, 4, 1-31.

DESCRIPTION:

A BASIC INSTRUCTIONAL PROGRAM IS BEING DEVELOPED AS A VEHICLE FOR RESEARCH IN TUTORIAL MODES OF COMPUTER-ASSISTED INSTRUCTION (CAI). SEVERAL DESIGN FEATURES WILL BE APPROPRIATE TO TRAINING IN OTHER TECHNICAL AREAS AND APPLICABLE IN OTHER INSTRUCTIONAL SETTINGS WHERE THE DEVELOPMENT OF ANALYTIC AND PROBLEM-SOLVING SKILLS IS A GOAL.

METHODS ARE INCORPORATED FOR MONITORING AND AIDING THE STUDENT AS HE WORKS ON PROGRAMMING PROBLEMS IN THE BASIC LANGUAGE. THE INSTRUCTIONAL PROGRAM DEVELOPED CAN BE USED TO INVESTIGATE SCHEMES FOR OPTIMIZING PROBLEM PRESENTATION AND GIVING ASSISTANCE DURING PROBLEM-SOLVING BASED ON A MODEL OF THE STUDENT'S ABILITIES AND DIFFICULTIES. PREVIOUS EXPERIENCE IN THE INSTRUCTIONAL AND TECHNICAL ASPECTS OF TEACHING A PROGRAMMING LANGUAGE INDICATES THAT A COURSE IN COMPUTER PROGRAMMING CAN BE DESIGNED TO HELP THE STUDENT ACQUIRE PROGRAMMING CONCEPTS IN A PERSONALIZED AND EFFICIENT MANNER AS HE DEVELOPS SKILLS AT INCREASINGLY ADVANCED LEVELS.

THIS ARTICLE REPORTS ON WORK CURRENTLY IN PROGRESS AND BRIEFLY SUMMARIZES OBSERVATIONS AND CONCLUSIONS BASED ON OPERATION DURING THE PILOT YEAR.

A MAJOR GOAL OF THE RESEARCH PROJECT IS TO INCREASE THE SOPHISTICATION WITH WHICH THE INSTRUCTIONAL PROGRAM MONITORS THE STUDENT'S WORK AND RESPONDS TO IT WITH APPROPRIATE HINTS AND PROMPTS. ONE ASPECT OF SUCH WORK IS THE UTILIZATION OF ALGORITHMS FOR CHECKING THE CORRECTNESS OF A STUDENT PROCEDURE. LIMITED BUT SUFFICIENT PROGRAM VERIFICATION IS POSSIBLE THROUGH SIMULATED EXECUTION OF THE PROGRAM ON TEST DATA STORED WITH EACH PROBLEM. WITHIN THE CONTROLLABLE CONTEXT OF INSTRUCTION, WHERE THE PROBLEMS TO BE SOLVED ARE PREDETERMINED AND THEIR SOLUTIONS KNOWN, SIMULATED EXECUTION OF THE STUDENT'S PROGRAM CAN EFFECTIVELY DETERMINE ITS CLOSENESS TO A STORED MODEL SOLUTION. (A)

17 COMPUTER-ASSISTED INSTRUCTION

BARR, A., BEARD, M., & ATKINSON, R.C. THE COMPUTER AS A TUTORIAL LABORATORY: THE STANFORD BIP PROJECT. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1976, 8, 567-596.

DESCRIPTION:

THE BASIC INSTRUCTIONAL PROGRAM (BIP) WAS DEVELOPED TO INVESTIGATE TUTORIAL MODES OF INTERACTION IN COMPUTER-ASSISTED INSTRUCTION. BIP IS AN INTERACTIVE PROBLEM-SOLVING LABORATORY THAT OFFERS TUTORIAL ASSISTANCE TO STUDENTS SOLVING INTRODUCTORY PROGRAMMING PROBLEMS IN THE BASIC LANGUAGE. THIS PAPER DESCRIBES HOW THE PROBLEM PRESENTATION SEQUENCE IS INDIVIDUALIZED BASED ON A REPRESENTATION OF THE STRUCTURE OF THE CURRICULUM AND A MODEL OF THE STUDENT'S STATE OF KNOWLEDGE. THE NATURE OF THE STUDENT-BIP INTERACTION IS CAPTURED IN AN ANNOTATED STUDENT DIALOGUE ILLUSTRATING A TYPICAL SESSION. (A)

32P, 20R.

18 CHIEF PROGRAMMER TEAMS

BARRY, B.S., & NAUGHTON, J.J. STRUCTURED PROGRAMMING SERIES (VOL. 10): CHIEF PROGRAMMER TEAM, OPERATIONS DESCRIPTION: FINAL REPORT (TECHNICAL REPORT RADC-TR-74-300-VOL-10). GRIFFISS AFB, NEW YORK: ROME AIR DEVELOPMENT CENTER, JANUARY 1975. (NTIS NO. AD AG08861)

DESCRIPTION:

THIS TECHNICAL REPORT CONTAINS JOB DESCRIPTIONS FOR EACH MEMBER OF A CHIEF PROGRAMMER TEAM. ALSO INCLUDED FOR BACKGROUND INFORMATION IS A DESCRIPTION OF THE CHIEF PROGRAMMER TEAM APPROACH TO SOFTWARE DEVELOPMENT, A DESCRIPTION OF THE TEAM COMPOSITION, AND MEMBER QUALIFICATIONS. (A)

53P, 5R.

19 DATA PROCESSING PERSONNEL TURNOVER

BARTOL, K.M. FACTORS RELATED TO EDP PERSONNEL COMMITMENT TO THE ORGANIZATION. COMPUTER PERSONNEL, 1977, 7(3), 2-5.

DESCRIPTION:

THIS PAPER INVESTIGATES THE RELATIONSHIP BETWEEN ORGANIZATIONAL COMMITMENT AND FIVE DIMENSIONS OF JOB SATISFACTION, AS WELL AS THE RELATIONSHIP BETWEEN COMMITMENT AND SEVERAL PERSONAL VARIABLES, FOR A SAMPLE OF EDP PERSONNEL. RESULTS SHOW A STRONG POSITIVE RELATIONSHIP BETWEEN JOB SATISFACTION AND ORGANIZATIONAL COMMITMENT FOR ALL FIVE SATISFACTION DIMENSIONS UNDER INVESTIGATION. SEVERAL PERSONAL VARIABLES, POSITION LEVEL, YEARS WORKING IN THE COMPUTER FIELD, AND INCOME, ALSO WERE SIGNIFICANTLY RELATED TO ORGANIZATIONAL COMMITMENT. (A)

20 SOFTWARE PHYSICS

BAYER, R. A THEORETICAL STUDY OF HALSTEAD'S SOFTWARE PHENOMENON (TECHNICAL REPORT NO. CSD-TR-69). LAFAYETTE, INDIANA: PURDUE UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, MAY 1972.

DESCRIPTION:

IN A RECENT PAPER HALSTEAD PRESENTED THE HYPOTHESIS THAT THE LENGTH OF A COMPUTER PROGRAM CAN BE DETERMINED FROM THE NUMBER OF DIFFERENT OPERATORS AND OPERANDS USED IN THE PROGRAM. IN THIS STUDY SEVERAL FORMAL MODELS OF PROGRAMS AND PROGRAMMING ARE PRESENTED. THE FIRST MODEL (R) YIELDS A PLAUSIBILITY ARGUMENT FOR HALSTEAD'S HYPOTHESIS AND THE BASIS FOR THE FOLLOWING MODELS (A,B,C). MODEL R SEEMS TOO DIFFICULT TO SOLVE, THUS MODELS A, B, AND C ARE DEVELOPED AS SUCCESSIVELY CLOSER APPROXIMATIONS TO MODEL R. THEY ARE SOLVED EXPLICITLY OR BY MONTE CARLO APPROXIMATIONS. THE RESULTS SUGGEST A POSSIBLE THEORETICAL EXPLANATION OF THE PHENOMENON OBSERVED BY HALSTEAD. (A)

20?, 1R.

21 COMPUTER-ASSISTED INSTRUCTION

BEARD, M.H., & BARR, A.V. THE BASIC INSTRUCTIONAL PROGRAM STUDENT MANUAL (SPECIAL REPORT 77-2). SAN DIEGO, CALIFORNIA: NAVAL PERSONNEL RESEARCH AND DEVELOPMENT CENTER, 1976.

DESCRIPTION:

THE BASIC INSTRUCTIONAL PROGRAM (BIP) IS A "HANDS-ON LABORATORY" THAT TEACHES ELEMENTARY PROGRAMMING IN THE BASIC LANGUAGE. THIS MANUAL IS THE STUDENT'S MAIN SOURCE OF INFORMATION ABOUT THE BIP SYSTEM AND THE BASIC LANGUAGE. THE MANUAL IS ORGANIZED AS A REFERENCE DOCUMENT AIMED AT STUDENTS WITH NO PREVIOUS PROGRAMMING EXPERIENCE. SECTION I INTRODUCES THE STUDENT TO THE COURSE ITSELF. SECTION II BEGINS WITH AN EXPLANATION OF PROGRAMMING IN GENERAL. DISCUSSIONS OF PROGRAMMING CONCEPTS SUCH AS INPUT AND VARIABLES ARE FOLLOWED BY THE SPECIFICATIONS OF THE BASIC STATEMENTS USED TO IMPLEMENT THEM. THE SYNTAX AND SAMPLE PROGRAMS ARE USED AS ILLUSTRATIONS. SECTION III LISTS AND EXPLAINS THE COMMANDS THAT CONTROL THE BIP SYSTEM. SOME ARE IDENTICAL TO STANDARD BASIC COMMANDS (E.G., RUN,LIST), AND OTHERS GIVE ACCESS TO THE UNIQUE FEATURES OF BIP. THE GLOSSARY LISTS ALL THE SPECIALIZED TERMS USED IN THE MANUAL, EXPLAINS THEIR USE BRIEFLY, AND GIVES REFERENCES TO THE SECTIONS WHERE DETAILED INFORMATION CAN BE FOUND. (A)

22 COMMENTS

BECKMAN, A. COMMENTS CONSIDERED HARMFUL. SIGPLAN NOTICES, APRIL 1977, 12(4), 94-96.

23 PROGRAM COMPLEXITY

BELL, D.E., & SULLIVAN, J.E. FURTHER INVESTIGATIONS INTO THE COMPLEXITY OF SOFTWARE (VOL. 2) (TECHNICAL REPORT NO. MTR-2874). BEDFORD, MASSACHUSETTS: MITRE CORP., 1975.

24 SOFTWARE ENGINEERING

BEMER, R.W. A SOFTWARE ENGINEER'S WORKSHOP: TOOLS AND TECHNIQUES. IN INFOTECH INFORMATION LTD., SOFTWARE ENGINEERING. BERKSHIRE, ENGLAND: INFOTECH INFORMATION LTD., 1972, 273-286.

DESCRIPTION:

THIS PAPER DESCRIBES THE PRODUCTION AIDS AND PRODUCTION METHODS THAT WERE INCORPORATED INTO A SOFTWARE FACTORY. THE SOFTWARE FACTORS ORIGINATED IN TWO PARTS: THE FIRST PART IS CONCERNED WITH ASSISTING THE PROGRAMMER; THE SECOND PART TRIES TO TAKE MEASURES OF PROGRAMMERS TO SEE IF ANY IMPROVEMENTS CAN BE MADE AND TO ENABLE STATISTICS TO BE DETERMINED. THE AIDS AND METHODS DISCUSSED IN THIS PAPER INCLUDE FILING SYSTEMS AND TEXT EDITING, DIRECTED GRAPH REPRESENTATIONS OF PROGRAMS, PRODUCTION SCHEDULING, AUTOMATIC FLOWCHARTING, "MAPPER," AND CERTIFICATION TECHNIQUES. EXAMPLES OF HOW THESE AIDS AND METHODS CAN BE USED TO IMPROVE PROGRAMMING PERFORMANCE ARE GIVEN. (MEA)

14P, CR.

25 PROGRAMMING, TASK ANALYSIS

BERGER, R.M. COMPUTER PROGRAMMER JOB ANALYSIS REFERENCE TEXT. MONTVALE, NEW JERSEY: AMERICAN FEDERATION OF INFORMATION PROCESSING SOCIETIES, 1974.

DESCRIPTION:

THIS REPORT PRESENTS THE RESULTS OF AN INVESTIGATION OF WHAT IN FACT PROGRAMMERS ARE AND DO. PROGRAMMING MANAGERS, SENIOR PROGRAMMERS, AND A CROSS SECTION OF PROGRAMMERS IN GENERAL PARTICIPATED IN THE DEVELOPMENT AND EVALUATION OF THE JOB DESCRIPTION. THESE THREE GROUPS, STARTING WITH MANAGEMENT, WERE GIVEN LISTS OF PROGRAMMING TASKS AND SKILLS TO EVALUATE. THE MANAGERS AND SENIOR PROGRAMMERS MADE REVISIONS AND ADDITIONS TO THE LISTS. FINALLY, PROGRAMMERS IN A NATIONWIDE SURVEY EVALUATED THE TASKS AND SKILLS FOR IMPORTANCE TO THEIR JOBS.

THE RESULTS OF THIS STUDY ARE EXPECTED TO BE USED BY MANAGEMENT PERSONNEL OF COMPUTER PROGRAMMING ORGANIZATIONS TO DEVELOP PROGRAMMER POSITION DESCRIPTIONS TO FIT THEIR SPECIFIC NEEDS. THE RESULTS ARE ALSO EXPECTED TO PROVIDE THE BASIS FOR DEVELOPMENT OF COMPREHENSIVE EXAMINATIONS FOR PROGRAMMER CERTIFICATION AND FOR DEVELOPMENT OF THE TRAINING REQUIRED FOR CERTIFICATION. FINALLY, THE STUDY SHOULD REPRESENT AN INITIAL MILESTONE IN THE CONTINUING PERSONNEL RESEARCH INTO THE NATURE AND EVOLUTION OF OCCUPATIONS IN THE COMPUTER FIELD. (A)

26 DEBUGGING

BERNSTEIN, W.A., & OWENS, J.T. DEBUGGING IN A TIME-SHARING ENVIRONMENT. AFIPS CONFERENCE PROCEEDINGS, 1968, 33, 7-14.

DESCRIPTION:

MODERN TIME-SHARING SYSTEMS HAVE RENDERED CONVENTIONAL MACHINE-LEVEL AND SYSTEM-LEVEL PROGRAM DEBUGGING AIDS OBSOLETE. A TIME-SHARING DEBUGGING SUPPORT SYSTEM IS PROPOSED FOR USE BY SYSTEM PROGRAMMERS IN THE DEBUGGING OF BOTH SYSTEM AND TASK PROGRAMS. A SET OF SUITABLE COMMANDS IS PROPOSED TO ALLOW BOTH ON-LINE DEBUGGING AND PROGRAMMING OF PREDEFINED ERROR PROCEDURES. SEVERAL PROBLEMS IN THE IMPLEMENTATION OF SUCH A SYSTEM ARE DISCUSSED AND POSSIBLE SOLUTIONS SUGGESTED. (HBR)

8P, CR.

27 PROGRAMMING METHODOLOGY

BIERMANN, A.W., & KRISHNASWAMY, R. CONSTRUCTING PROGRAMS FROM EXAMPLE COMPUTATIONS. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1976, SE-2, 141-153.

28 STRUCTURED PROGRAMMING

BIGELOW, R. STRUCTURED CODE VIA STACK MACHINE. COMPUTER, JUNE 1975, 8(6), 67.

29 SOFTWARE DESIGN

BLACK, U.D. PSYCHOLOGY APPLIED TO SOFTWARE DESIGN. INFOSYSTEMS, MARCH 1978, PP. 86; 90-91.

DESCRIPTION:

DURING THE PAST FEW YEARS THE DATA PROCESSING INDUSTRY HAS MADE SIGNIFICANT PROGRESS IN PROVIDING A THEORETICAL FOUNDATION FOR SOFTWARE DEVELOPMENT. THE USE OF MATHEMATICAL CONCEPTS TO DESCRIBE THE PROGRAMMING PROCESS HAS CONTRIBUTED TO THIS PROGRESS. THE INDUSTRY, HOWEVER, HAS MADE LESS PROGRESS IN DEVELOPING A FOUNDATION FOR UNDERSTANDING PROCESSES INVOLVED IN THE MORE SUBJECTIVE ASPECTS OF SOFTWARE DEVELOPMENT. I REFER HERE TO THE COGNITIVE, OR THOUGHT, PROCESSES. CERTAIN IDEAS, NOTABLY STRUCTURED PROGRAMMING AND STRUCTURED DESIGN, ADVANCED BY SOME OF THE MATHEMATICAL THEORISTS ARE SUBSTANTIATED BY PRINCIPLES OF PSYCHOLOGY. TO ILLUSTRATE MY POINT, I HAVE CHOSEN A SCHOOL OF THOUGHT IN PSYCHOLOGY BROADLY REFERRED TO AS GESTALTISM. (A, ABBR.)

3P, 2R.

30 SOFTWARE DESIGN

BOEHM, B.W. SOFTWARE DESIGN AND STRUCTURING. IN E. HOROWITZ (ED.) PRACTICAL STRATEGIES FOR DEVELOPING LARGE SOFTWARE SYSTEMS. READING, MASS.: ADDISON-WESLEY, 1975, 103-128.

DESCRIPTION:

THIS PAPER PROVIDES A CLASSIFICATION OF SOFTWARE DESIGN AND STRUCTURING TECHNIQUES INTO A FEW MAIN CATEGORIES WHICH CAN CURRENTLY BE CONSIDERED AS SOFTWARE DESIGN ALTERNATIVES. FOR EACH ALTERNATIVE, IT PRESENTS A SHORT DESCRIPTION AND A BALANCE SHEET INDICATING THE ADVANTAGES AND DIFFICULTIES IN USING THE TECHNIQUE.

IN AN OVERALL COMPARISON OF THE ALTERNATIVE TECHNIQUES, NONE OF THEM PROVIDES POSITIVE ASSURANCE OF SATISFYING ALL THE CRITERIA, BUT EACH CRITERION IS SATISFIED BY AT LEAST ONE OF THE TECHNIQUES. THIS WOULD LEAD ONE TO BELIEVE THAT AN APPROPRIATE SYNTHESIS OF THE TECHNIQUES MIGHT PROVE SUCCESSFUL.

THE FOLLOWING MAJOR DESIGN AND STRUCTURING ALTERNATIVES ARE DESCRIBED AND EVALUATED: BOTTOM-UP, TOP-DOWN STUB, TOP-DOWN/PROBLEM STATEMENT, STRUCTURED PROGRAMMING, AND MODEL-DRIVEN DESIGN. (A)

26P, 22R.

31 SOFTWARE ENGINEERING

BOEHM, B.W., MCCLEAN, R.K., & URFRIG, D.B. SOME EXPERIENCE WITH AUTOMATED AIDS TO THE DESIGN OF LARGE-SCALE RELIABLE SOFTWARE. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 125-133 (ALSO PROCEEDINGS, INTERNATIONAL CONFERENCE ON RELIABLE SOFTWARE, 21-23 APRIL 1975, LOS ANGELES, CALIFORNIA. SIGPLAN NOTICES, JUNE 1975, 10(6), 105-113).

DESCRIPTION:

THIS PAPER SUMMARIZES SOME RECENT EXPERIENCE IN ANALYZING AND ELIMINATING SOURCES OF ERROR IN THE DESIGN PHASE OF LARGE SOFTWARE PROJECTS. IT BEGINS BY POINTING OUT SOME OF THE SIGNIFICANT DIFFERENCES IN SOFTWARE ERROR INCIDENCE BETWEEN LARGE AND SMALL SOFTWARE PROJECTS. THE MOST STRIKING CONTRAST, ILLUSTRATED BY PROJECT DATA, IS THE LARGE PREPONDERANCE OF DESIGN ERRORS OVER CODING ERRORS ON LARGE-SCALE PROJECTS, NOT ONLY WITH RESPECT TO NUMBERS OF ERRORS, BUT ALSO WITH RESPECT TO THE RELATIVE TIME AND EFFORT REQUIRED TO DETECT THEM AND CORRECT THEM.

THE PAPER NEXT PRESENTS A TAXONOMY OF SOFTWARE ERROR CAUSES, AND SOME ANALYSES OF THE DESIGN ERROR DATA, PERFORMED TO OBTAIN A BETTER UNDERSTANDING OF THE NATURE OF LARGE-SCALE SOFTWARE DESIGN ERRORS AND TO EVALUATE ALTERNATIVE METHODS OF PREVENTING, DETECTING AND ELIMINATING THEM.

BASED ON THIS ANALYSIS OF OBSERVATIONAL DATA, A HYPOTHESIS WAS DERIVED REGARDING THE POTENTIAL COST-EFFECTIVENESS OF AN AUTOMATED AID TO DETECTING INCONSISTENCIES BETWEEN ASSERTIONS ABOUT THE NATURE OF INPUTS AND OUTPUTS OF THE VARIOUS ELEMENTS (FUNCTIONS, MODULES, DATA BASES, DATA SOURCES, ETC.) OF THE SOFTWARE DESIGN. THIS HYPOTHESIS WAS TESTED BY DEVELOPING A PROTOTYPE VERSION OF SUCH AN AID, THE DESIGN ASSERTION CONSISTENCY CHECKER (DACC), USING TRW'S GENERALIZED INFORMATION MANAGEMENT (GIM) SYSTEM, AND USING IT ON A LARGE-SCALE SOFTWARE PROJECT WITH 186 ELEMENTS AND 967 ASSERTIONS ABOUT THEIR INPUTS AND OUTPUTS.

IN GENERAL, THE DATA CONFIRMED THE HYPOTHESIS ABOUT THE GENERAL UTILITY OF A DACC CAPABILITY FOR LARGE SOFTWARE PROJECTS. HOWEVER, A NUMBER OF ADDITIONAL FEATURES SHOULD BE CONSIDERED TO COMPENSATE FOR CURRENT DEFICIENCIES (IN AREAS SUCH AS MANUSCRIPT PREPARATION) AND TO FULLY TAKE ADVANTAGE OF HAVING THE SOFTWARE DESIGN IN MACHINE-READABLE FORM. (A) 9P, 16R.

32 PROGRAMMING LANGUAGES

BOEHM, H.-P., FISCHER, H.L., & RAULEFS, P. CSSA: LANGUAGE CONCEPTS AND PROGRAMMING METHODOLOGY. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 100-108 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 100-108).

33 STATISTICS ON USE OF TIME-SHARING SYSTEM

BOIES, S.J. USER BEHAVIOR ON AN INTERACTIVE COMPUTER SYSTEM. IBM SYSTEMS JOURNAL, 1974, 13, 2-12.

DESCRIPTION:

THE USER-COMPUTER SYSTEM INTERACTIONS DESCRIBED HERE ARE CONFINED TO IBM/360 TIME SHARING SYSTEM (TSS) AND ITS TERMINALS. THE VARIOUS FACTORS DISCUSSED ARE: THE DURATION AND FREQUENCY OF TERMINAL SESSIONS, THE USER OF LANGUAGE PROCESSORS, COMMAND USAGE AND USER RESPONSE TIME. THE DATA INDICATE THAT A VERY SMALL PERCENTAGE OF USERS ACCOUNT FOR A LARGE PERCENTAGE OF THE TOTAL TERMINAL USAGE. USERS SELDOM USE THE ERROR CORRECTION FEATURES OF THE LANGUAGE PROCESSOR EVEN IF THEY NEED THEM. A LONG SYSTEM RESPONSE TIME IS RELATED TO A LONG USER RESPONSE TIME. ONLY A SMALL NUMBER OF COMMANDS ACCOUNT FOR A LARGE FREQUENCY OF USAGE. WHEN ONE COMMAND WOULD HAVE BEEN SUFFICIENT SEVERAL COMMANDS ARE OFTEN USED BY THE USERS. (O) 17P, 10R.

34 PROGRAMMING ERRORS

BOIES, S.J., & GOULD, J.D. SYNTACTIC ERRORS IN COMPUTER PROGRAMMING. HUMAN FACTORS, 1974, 16, 253-257.

DESCRIPTION:

A STUDY OF USERS OF A LARGE-SCALE COMPUTER SYSTEM (TSS/360) REVEALED THAT ONLY 12 TO 17% OF THE FORTRAN, PL/I AND ASSEMBLER LANGUAGE COMPUTER PROGRAMS SUBMITTED TO THE LANGUAGE PROCESSORS CONTAINED SYNTACTIC ERRORS. THUS, SYNTACTIC ERRORS DO NOT APPEAR TO BE A SIGNIFICANT BOTTLENECK IN PROGRAMMING. THIS EXPERIMENT IS PART OF A LARGER EFFORT TO IDENTIFY AND REDUCE THE BEHAVIORAL BOTTLENECKS IN COMPUTER PROGRAMMING. (A)
5P, 11R.

35 STATISTICS ON USE OF INTERACTIVE DEBUGGING FACILITIES

BOIES, S.J., & SPIEGEL, M.F. A BEHAVIORAL ANALYSIS OF PROGRAMMING: ON THE USE OF INTERACTIVE DEBUGGING FACILITIES (TECHNICAL REPORT RC-4472). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, AUGUST 1973. (NTIS NO. AD 772127)

DESCRIPTION:

MEASUREMENT AND ANALYSIS OF USER DEBUGGING BEHAVIOR ON PRESENT INTERACTIVE SYSTEMS CAN LEAD TO AN UNDERSTANDING OF HOW FUTURE INTERACTIVE SYSTEMS SHOULD BE IMPLEMENTED. THIS STUDY PRESENTS BOTH AN ANALYSIS OF THE ON-LINE USAGE OF THE TSS/360 PROGRAM CONTROL SYSTEM (PCS) AS WELL AS THE RESULTS OF A QUESTIONNAIRE DESIGNED TO PROBE THE PCS KNOWLEDGE OF THE POPULATION OF USERS OF THAT SYSTEM. THE QUESTIONNAIRE RESPONSE POPULATION TENDED TO BE DIVIDED INTO THREE COMPETENCE GROUPS, THE HIGHEST OF WHICH CONSISTED MOSTLY OF SYSTEM SUPPORT PERSONNEL AND ASSEMBLER LANGUAGE PROGRAMMERS. THE ON-LINE USAGE ANALYSIS SHOWED THAT PCS COMMANDS ACCOUNTED FOR 7.2 PERCENT OF ALL COMMANDS ENTERED BY PROGRAMMERS. PCS WAS USED RARELY TO MODIFY THE LOGIC OF A PROGRAM, BUT WAS USED AS A PROGRAM VARIABLE INPUT/OUTPUT SYSTEM. IT WAS CONCLUDED THAT THE UNDERSTANDING OF THE INTERACTIVE DEBUGGING PROCESS MUST COME FROM THE PERSPECTIVE OF THE ENTIRE SYSTEM, AS UP TO 50 PERCENT OF THE DEBUGGING EFFORT IS SPENT ON TASK AND DATA MANAGEMENT. FIVE CRITICAL FEATURES OF AN EFFECTIVE INTERACTIVE PROGRAMMING SYSTEM ARE OUTLINED. (A)
25P, 12R.

36 QUERY LANGUAGE

BOYCE, R.F., CHAMBERLIN, D.D., KING, W.F., III, & HAMMER, M.M. SPECIFYING QUERIES AS RELATIONAL EXPRESSIONS: THE SQUARE DATA SUBLANGUAGE. COMMUNICATIONS OF THE ACM, 1975, 18, 621-628.

DESCRIPTION:

THIS PAPER PRESENTS A DATA SUBLANGUAGE CALLED SQUARE, INTENDED FOR USE IN AD HOC, INTERACTIVE PROBLEM SOLVING BY NON-COMPUTER SPECIALISTS. SQUARE IS BASED ON THE RELATIONAL MODEL OF DATA, AND IS SHOWN TO BE RELATIONALLY COMPLETE. HOWEVER, IT AVOIDS THE QUANTIFIERS AND BOUND VARIABLES REQUIRED BY LANGUAGES BASED ON THE RELATIONAL CALCULUS. FACILITIES FOR QUERY, INSERTION, DELETION, AND UPDATE ON TABULAR DATA BASES ARE DESCRIBED. A SYNTAX IS GIVEN, AND SUGGESTIONS ARE MADE FOR ALTERNATIVE SYNTAXES, INCLUDING A SYNTAX BASED ON ENGLISH KEY WORDS FOR USERS WITH LIMITED MATHEMATICAL BACKGROUND. (A)
8P, 20R.

37 TURNOVER AMONG DATA PROCESSING PERSONNEL

BRADFORD, P.A., & COTTRELL, L.R. FACTORS INFLUENCING BUSINESS DATA PROCESSORS TURNOVER: A COMPARATIVE CASE HISTORY. COMPUTER PERSONNEL, 1977, 7(1-2), 3-6.

38 SOFTWARE ENGINEERING

BRATMAN, H., & COURT, T. THE SOFTWARE FACTORY. COMPUTER MAGAZINE, MAY 1975, 8(5), 28-37.

DESCRIPTION:

THIS PAPER DESCRIBES AN EFFORT TO DEVELOP AN INTEGRATED SET OF SOFTWARE DEVELOPMENT TOOLS TO SUPPORT A DISCIPLINED AND REPEATABLE APPROACH TO SOFTWARE DEVELOPMENT. THIS EFFORT, WHICH IS REALLY PART OF A LARGER PROGRAM CURRENTLY UNDERWAY AT SYSTEM DEVELOPMENT CORPORATION TO INCREASE SOFTWARE RELIABILITY AND CONTROL SOFTWARE PRODUCTION COSTS, IS INTENDED TO REPLACE WHAT THE AUTHORS TERM "AD HOC CONGLOMERATIONS OF DEVELOPMENTAL TOOLS" WITH STANDARD ENGINEERING TECHNIQUES. (O)

10P, 7R.

39 PROGRAMMING

BRATMAN, H., MARTIN, H.G., & PERSTEIN, E.C. PROGRAM COMPOSITION AND EDITING WITH AN ONLINE DISPLAY. AFIPS CONFERENCE PROCEEDINGS, 1968, 33, 1349-1360.

DESCRIPTION:

AN INTERACTIVE PROGRAMMING SUPPORT SYSTEM (IPSS) HAS BEEN UNDER DEVELOPMENT AT SYSTEM DEVELOPMENT CORPORATION SINCE 1965. THE PURPOSE OF THE SYSTEM IS TO PERMIT ALL OF THE PROGRAMMING PROCESSES -- COMPOSITION (IN A PROCEDURE-ORIENTED LANGUAGE), EDITING, EXECUTION, TESTING, AND DOCUMENTATION -- TO BE CARRIED OUT AS PARTS OF A SINGLE, COORDINATED ACTIVITY CENTERED AROUND AN INTERACTIVE COMPILER. IPSS ATTEMPTS TO UNIFY TECHNIQUES THAT ARE USUALLY EMBODIED IN SEPARATE FUNCTIONAL PROGRAMS, SO THAT THE PROGRAMMER NEED NOT KNOW WHICH PARTICULAR PROGRAM IS PERFORMING A SPECIFIC TASK. THE SYSTEM IS INTENDED FOR A TIME-SHARING ENVIRONMENT, WITH USER INTERACTION VIA A SMALL TABULAR DISPLAY OR TYPEWRITER-LIKE TERMINAL. (A)

12P, 4R.

40 SOFTWARE ENGINEERING

BROOKS, F.P., JR. THE MYTHICAL MAN-MONTH: ESSAYS ON SOFTWARE ENGINEERING. READING, MASSACHUSETTS: ADDISON-WESLEY, 1975.

DESCRIPTION:

THIS BOOK CONTAINS FIFTEEN ESSAYS ON THE MANAGEMENT OF COMPUTER PROGRAMMING PROJECTS. THESE ESSAYS PRIMARILY DESCRIBE THE AUTHOR'S PERSONAL OPINIONS AND EXPERIENCES BUT ALSO PROVIDE A BRIEF INTRODUCTION TO THE LITERATURE ON SOFTWARE ENGINEERING. THE CENTRAL THESIS IS THAT LARGE SOFTWARE PROJECTS SUFFER FROM PROBLEMS THAT ARE QUALITATIVELY DIFFERENT FROM THOSE ENCOUNTERED IN SMALL PROJECTS DUE TO THE DIVISION OF LABOR. FOR THIS REASON, THE MOST IMPORTANT GOAL IN A LARGE SOFTWARE PROJECT IS TO MAINTAIN CONCEPTUAL INTEGRITY. THESE ESSAYS FOCUS BOTH ON THE DIFFICULTIES IN ACHIEVING CONCEPTUAL INTEGRITY AND ON METHODS FOR ACHIEVING IT. (MEA)

203P, 83R.

41 PROGRAMMING

BROOKS, R. A MODEL OF HUMAN COGNITIVE BEHAVIOR IN WRITING CODE FOR COMPUTER PROGRAMS (2 VOLS.). PITTSBURGH, PENNSYLVANIA: CARNEGIE-MELLON UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, MAY 1975 (ALSO: REPORTED BRIEFLY IN PROCEEDINGS OF THE FOURTH INTERNATIONAL JOINT CONFERENCE OF ARTIFICIAL INTELLIGENCE, 1975). (NTIS NOS. AD A013582, AD A012918)

DESCRIPTION:

A THEORY OF HUMAN COGNITIVE PROCESSES IN WRITING CODE FOR COMPUTER PROGRAMS IS PRESENTED. IT VIEWS BEHAVIOR IN TERMS OF THREE PROCESSES, UNDERSTANDING, PLANNING, AND CODING. THE FIRST OF THESE CONSISTS OF ACQUISITION OF INFORMATION FROM THE PROBLEM INSTRUCTIONS AND DIRECTIONS. THIS IS USED BY THE PLANNING PROCESS TO CREATE A SOLUTION PLAN STATED AS A SET OF FUNCTIONAL SPECIFICATIONS IN A LANGUAGE WHICH IS INDEPENDENT OF THE SYNTAX OF THE PARTICULAR PROGRAMMING LANGUAGE. THE CODING PROCESS CONVERTS THIS PLAN TO CODE USING A PROCESS NAMED "SYMBOLIC EXECUTION" IN WHICH THE PIECES OF CODE ARE ASSIGNED EFFECTS IN TERMS OF THE FUNCTIONS THE PROGRAMMER INTENDS THE CODE TO PERFORM IN ACHIEVING THE PURPOSE OF THE PROGRAM.

WITHIN THE FRAMEWORK OF THIS THEORY, A MORE EXPLICIT MODEL OF THE CODING PROCESS WAS DEVELOPED. THE MODEL IS BASED ON A PRODUCTION SYSTEM AND HAS BEEN IMPLEMENTED AS A COMPUTER PROGRAM. GIVEN PLANS TAKEN FROM PROTOCOLS OF A PROGRAMMER WRITING A SERIES OF SHORT FORTRAN PROGRAMS, IT IS ABLE TO GENERATE THE SAME CODE IN THE SAME ORDER AS THE PROGRAMMER DID.

THE MODEL MAKES THREE ASSUMPTIONS ABOUT PROGRAMMER BEHAVIOR IN WRITING PROGRAMS:

1. PROGRAMMERS HAVE A LARGE AMOUNT OF SPECIFIC KNOWLEDGE ABOUT HOW TO ENCODE PARTICULAR PLAN ELEMENTS.
2. PROGRAMMERS GENERATE CODE BY USING THE EFFECTS ASSIGNED TO EACH PIECE TO GENERATE THE NEXT.
3. THE BASIC UNITS OF A PROGRAMMER'S KNOWLEDGE OF LANGUAGE SYNTAX ARE DETERMINED BY THE WAY IN WHICH HE USES THE LANGUAGE, RATHER THAN BY PROPERTIES OF THE SYNTAX ALONE.

THE IMPLICATIONS OF THESE ASSERTIONS ARE DISCUSSED FOR THE USE OF PRODUCTION SYSTEMS TO REPRESENT BEHAVIOR, FOR TEACHING PROGRAMMING, FOR ERROR ANALYSIS IN DEBUGGING, AND FOR THE USE OF BACK-TRACKING IN PROBLEM SOLVING SYSTEMS. (A)

304P, 46R.

42 PROGRAMMING

BROOKS, R. HOW A PROGRAMMER UNDERSTANDS A PROGRAM: A MODEL (TECHNICAL REPORT NO. 97). IRVINE, CALIFORNIA: UNIVERSITY OF CALIFORNIA, DEPARTMENT OF INFORMATION AND COMPUTER SCIENCE, 1977.

DESCRIPTION:

IN A LARGE VARIETY OF PROGRAMMING SITUATIONS, A PROGRAMMER IS REQUIRED TO UNDERSTAND A PROGRAM THAT SOMEONE ELSE HAS WRITTEN. A MODEL HAS BEEN CREATED FOR THE BEHAVIOR SEEN IN THE VERBAL PROTOCOL OF A PROGRAMMER ON A SAMPLE UNDERSTANDING TASK. THE MODEL IS BASED ON A THEORY OF UNDERSTANDING WHICH STRESSES THE ROLE OF THE PROGRAMMER'S A PRIORI HYPOTHESES OR GUESSES ABOUT THE PROGRAM STRUCTURE. ORGANIZATION OF THE MODEL IS THAT OF A PRODUCTION SYSTEM, A STRUCTURE WHICH APPEARS PARTICULARLY WELL-SUITED TO THE ASYNCHRONOUS, NON-SEQUENTIAL NATURE OF THE INPUT. (A)

34P, 9R.

43 PROGRAMMING

BROPHY, H.F. IMPROVING PROGRAMMING PERFORMANCE. AUSTRALIAN COMPUTER JOURNAL, 1970, 2(2), 66-70.

DESCRIPTION:

THERE IS RELATIVELY LITTLE WRITTEN OR DISCUSSED ABOUT PROGRAMMERS, WHO REPRESENT A LARGE PART OF THE INVESTMENT IN COMPUTING. THIS PAPER ATTEMPTS TO FOCUS ATTENTION ON AN UNDERSTANDING AND EVALUATION OF THE PROGRAMMING TASK, SUGGESTING WAYS TO IMPROVE THE PROGRAMMER'S LOT, AND TO RAISE HIS PRODUCTIVITY. IT IS IMPORTANT TO REALIZE THAT THE PROGRAMMER IS INVOLVED IN HUMAN PROBLEM SOLVING, AND COMMUNICATING NOT ONLY WITH THE MACHINE, BUT WITH OTHER PEOPLE. HIS ULTIMATE PRODUCT WILL REFLECT HIS CREATIVITY, EFFICIENCY AND ABILITY TO COMMUNICATE.

TO GET MORE FROM THE PROGRAMMER, WE SHOULD PROVIDE HIM WITH BETTER TOOLS, TRAIN HIM IN THE USE OF THOSE TOOLS, AND STANDARDIZE AND MANAGE HIS PERFORMANCE. SEVERAL SPECIFIC IDEAS AND SUGGESTIONS ARE GIVEN, MANY OF THEM HAVING BEEN ALREADY INSTITUTED IN A FEW ORGANIZATIONS. (A)

5P, 19R.

44 PROGRAMMING

BROWN, G.D. PROGRAMMING: THE QUIET EVOLUTION. DATAMATION, 1972, 18, 147-150.

DESCRIPTION:

SOFTWARE, UNLIKE HARDWARE, DOES NOT ADVANCE IN CLEAR-CUT GENERATIONS. ITS CHANGES DERIVE MORE FROM HUMAN NEEDS THAN FROM TECHNOLOGICAL POSSIBILITIES. THE EFFECTS OF CIVIL DISORDER, VIET NAM, THE SPACE PROGRAM, AND TRENDS OF THE ECONOMY AND RESEARCH ARE MENTIONED AS INFLUENCES ON THE EVOLUTION OF PROGRAMMING. FOR THE FORSEEABLE FUTURE, PROGRAMMING WILL REMAIN AN EXACTING SKILL. NO ECONOMIES OF SCALE ARE YET APPARENT, AND AUTOMATIC OR MASS PROGRAMMING SEEMS AS FAR AWAY AS EVER. THE PROGRAMMING PROFESSION WILL REQUIRE MORE EDUCATION, PRIDE IN COMPETENCE, AND A PREOCCUPATION WITH QUALITY. (GDC)

4P, 0R.

45 MAN-MACHINE SYSTEM MODELING

BROWN, J.F., JR., CAGLIA, W.E., & SEITLE, R.A. THE USE OF MAN/MACHINE INTERACTION MODELS IN SHORTENING SYSTEM DEVELOPMENT CYCLES. IN PROCEEDINGS OF THE FIFTH NATIONAL SYMPOSIUM ON HUMAN FACTORS IN ELECTRONICS. NEW YORK: INSTITUTE OF ELECTRICAL AND ELECTRONIC ENGINEERS, 1964, 304-313.

46 STRUCTURED PROGRAMMING

BROWN, J.R. STRUCTURED PROGRAMMING: AGONY AND/OR ECSTACY. COMPUTER, JUNE 1975, 8(6), 56-57.

47 SOFTWARE DEVELOPMENT

BROWN, P.J. PROGRAMMING AND DOCUMENTING SOFTWARE PROJECTS. COMPUTING SURVEYS, 1974, 6, 213-220.

DESCRIPTION:

TIME AND TIME AGAIN SOFTWARE PROJECTS, THOUGH UNDERTAKEN BY PEOPLE OF CONSIDERABLE INTELLECTUAL ABILITY, FAIL. THIS WILL DOUBTLESS BE A CONTINUING PHENOMENON, BECAUSE SOFTWARE PRODUCTION IS INDEED A DIFFICULT TASK REQUIRING WIDE-RANGING SKILLS. CERTAINLY, NO SINGLE PAPER CAN SUDDENLY SOLVE ALL THE PROBLEMS OF SOFTWARE WRITING AND ELIMINATE THE FAILURES. THE SCOPE OF THIS PAPER IS LIMITED TO FOUR STAGES IN THE EXECUTION OF SOFTWARE PROJECTS, NAMELY THE PLANNING, CODING, TESTING, AND FINAL USAGE. THESE FOUR CRUCIAL AREAS CONTAIN PITFALLS THAT HAVE BEEN RESPONSIBLE FOR A LARGE PROPORTION OF SOFTWARE FAILURES, AND IT IS HOPED THIS PAPER WILL HELP AVOID SOME OF THEM. (A)

8P, 16R.

43 TIME-SHARING

BROWN, T., & KLERER, M. THE EFFECT OF LANGUAGE DESIGN ON TIME-SHARING OPERATIONAL EFFICIENCY. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1975, 7, 233-247.

DESCRIPTION:

THE IMPORTANCE OF 'THINK TIME' FOR OPERATIONAL EFFICIENCY OF TIME-SHARING SYSTEMS IS RE-EMPHASIZED. IT IS POINTED OUT THAT THINK TIME IS AN EXPERIMENTAL DEPENDENT PARAMETER OF THE SOFTWARE-HARDWARE PROGRAMMING SYSTEM AND MAY BE LENGTHENED OR SHORTENED AS A FUNCTION OF CONSOLE OR PROGRAMMING LANGUAGE DESIGN. A SIMPLE COMPUTATIONAL MODEL IS USED TO PREDICT THE BEHAVIOR OF RESPONSE TIME AS A FUNCTION OF THINK TIME FOR DIFFERENT CONDITIONS OF SERVICE LOADING. THE ECONOMIC IMPLICATIONS ARE CONSIDERED. (A)

15P, 13R.

49 PROGRAMMING LANGUAGES

BRUNT, R.F., & TUFFS, D.E. A USER-ORIENTED APPROACH TO CONTROL LANGUAGES. SOFTWARE-PRACTICE AND EXPERIENCE, 1976, 6, 93-108.

DESCRIPTION:

THIS PAPER DESCRIBES THE DESIGN APPROACH ADOPTED FOR SCL WHICH IS THE CONTROL LANGUAGE OF SYSTEM B, THE OPERATING SYSTEM OF ICL'S NEW 2900 SERIES OF COMPUTERS. THE DESIGN EMPHASIS OF SCL IS 'USABILITY' AND THE PAPER SETS OUT TO SHOW THAT SCL PROVIDES WHAT USERS REQUIRE.

THE AUTHORS BOTH WORK IN THE SYSTEMS PROGRAMMING DIVISION OF ICL WHERE THE MAJORITY OF THEIR RECENT WORK HAS CENTERED AROUND THE DESIGN OF JOB MANAGEMENT IN SYSTEM B AND SCL IN PARTICULAR. (A)

16P, 4R.

50 STATISTICS ON USE OF A TIME-SHARING SYSTEM

BRYAN, G.E. JOSS: 20,000 HOURS AT A CONSOLE -- A STATISTICAL SUMMARY. AFIPS CONFERENCE PROCEEDINGS, 1967, 31, 769-777 (ALSO TECHNICAL REPORT NO. RM-5359-PR, SANTA MONICA, CALIFORNIA: RAND CORPORATION, AUGUST 1967).

DESCRIPTION:

JOSS IS A SPECIAL PURPOSE INTERACTIVE COMPUTATIONAL FACILITY. THIS PAPER REPORTS ON EFFORTS TO MEASURE SYSTEM USE AND CHARACTERISTICS OF INDIVIDUAL SYSTEM USERS. THIS ALLOWS THE ASSESSMENT OF HOW WELL THE SYSTEM MEETS THE DEMANDS OF A "TYPICAL" USER. (MEA)

9P, 23R.

51 SOFTWARE DEVELOPMENT

BULLEN, R.H., JR. ENGINEERING OF QUALITY SOFTWARE SYSTEMS (SOFTWARE FIRST CONCEPTS) (TECHNICAL REPORT NO. MTR-2648-VOL-3). BEDFORD, MASSACHUSETTS: MITRE CORP., JANUARY 1975.

52 SOFTWARE PHYSICS

BULUT, N. INVARIANT PROPERTIES IN ALGORITHMS. UNPUBLISHED DOCTORAL DISSERTATION, PURDUE UNIVERSITY, LAFAYETTE, INDIANA, 1973.

53 SOFTWARE PHYSICS

BULUT, N., & HALSTEAD, M.H. IMPURITIES FOUND IN ALGORITHM IMPLEMENTATIONS (TECHNICAL REPORT NO. CSD-TR-111). LAFAYETTE, INDIANA: PURDUE UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, 1974.

DESCRIPTION:

THIS PAPER DESCRIBES SIX IMPURITIES OBTAINED ON THE BASIS OF CONFORMITY TO THE SOFTWARE PHYSICS RELATIONS, AS FOLLOWS: (1) SELF-CANCELLING OPERATIONS, (2) AMBIGUOUS USAGE OF AN OPERAND, (3) SYNONYMOUS USAGES OF OPERANDS, (4) COMMON SUBEXPRESSIONS, (5) UNNECESSARY REPLACEMENTS, AND (6) UNFACTORED EXPRESSIONS. THE IMPURITIES WERE MOSTLY APPARENT IN STUDENT PROGRAMS, NOT IN PUBLISHED ONES. (C)

4P, 11R.

54 SOFTWARE PHYSICS

BULUT, N., HALSTEAD, M.H., & BAYER, R. EXPERIMENTAL VALIDATION OF A STRUCTURAL PROPERTY OF FORTRAN ALGORITHMS (TECHNICAL REPORT NO. CSD-TR-115). LAFAYETTE, INDIANA: PURDUE UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, 1974.

DESCRIPTION:

THE CORRELATION BETWEEN OBSERVED AND CALCULATED LENGTH FOR 429 FORTRAN PROGRAMS IN THE PROGRAM LIBRARY AT PURDUE UNIVERSITY WAS .95, WITH PROGRAM SIZE RANGING FROM 3 TO 1674 STATEMENTS. THE STUDY CONFIRMS THE EXISTENCE OF A FUNCTIONAL RELATIONSHIP BETWEEN THE MEASURABLE PARAMETERS N(1), N(2), AND N.

THIS PAPER ALSO REVIEWS RULES OBSERVED IN COUNTING ALGORITHM. (O)

8P, 10R.

55 SOFTWARE ENGINEERING

BURNS, I.F., HANSING, M.M., HERRING, F.P., & MCCOY, R.C. SOFTWARE ENGINEERING AND SPECIFICATION VALIDATION. COMPUTER-AIDED SOFTWARE ENGINEERING PROGRAM, SOFTWARE CAPABILITIES DESCRIPTION (TECHNICAL REPORT NO. TRW-22944-6921-006). HUNTSVILLE, ALABAMA: TRW SYSTEMS GROUP, JANUARY 1974. (NTIS NO. AD 915757)

DESCRIPTION:

THIS REPORT DESCRIBES THE CAPABILITIES OF THE COMPUTER-AIDED SOFTWARE ENGINEERING PROGRAM (CASEP), TO BE DEVELOPED BY TRW UNDER THE SPONSORSHIP OF ABMDA.

CASEP IS AN INTEGRATED SET OF PROGRAMS WHICH ARE DESIGNED TO ENHANCE THE CURRENT ABMDA SOFTWARE ENGINEERING METHODOLOGY. THIS ENHANCEMENT IS ACCOMPLISHED BY ASSURING COMPLETENESS OF SPECIFICATION OF SOFTWARE DESIGN, BY ASSURING CORRECTNESS OF SPECIFIED LOGIC IN PERFORMING STATIC VALIDATION AT ALL LEVELS OF DEVELOPMENT, BY AIDING THE DEVELOPMENT OF SIMULATORS TO VALIDATE THE SPECIFICATIONS, BY PROVIDING ASSISTANCE TO THE PROCESS DESIGNER DURING THE PROCESS DESIGN PHASE, AND BY PROVIDING MANAGEMENT INFORMATION REPORTS AND CONFIGURATION MANAGEMENT CONTROLS AS AN INTEGRAL PART OF THE DEVELOPMENT SUPPORT. (A)

75P, 27R.

56 PROGRAMMING STYLE

BYARS, H.E. AN INVESTIGATION INTO PROGRAMMING STYLE. COMPUTER STUDIES IN THE HUMANITIES AND VERBAL BEHAVIOR, 1969, 2, 198-203.

DESCRIPTION:

THIS PAPER SEEKS TO ANSWER THE QUESTION: WHAT IS A GOOD BASIS FOR DETERMINING WHO WROTE A GIVEN PROGRAM? ANALYSIS OF PROGRAMMING STYLE PROCEEDS ON TWO LEVELS (PROGRAM TEXT AND PROGRAM SEMANTICS) IN ORDER TO IDENTIFY STYLISTIC DISCRIMINATORS AMONG PROGRAMS OF VARYING AUTHORSHIP. BECAUSE OF ITS FLEXIBILITY AND ITS EASE OF PARSING, THE STUDY CONCENTRATED ON THE FORTRAN LANGUAGE. (A)

6P, 2R.

57 SOFTWARE DESIGN

CAMERON, M.D. HUMAN PROBLEMS IN SOFTWARE DESIGN. IN PROCEEDINGS OF THE TENTH ANNUAL MEETING, HUMAN FACTORS ASSOCIATION OF CANADA. DOWNSVIEW, ONTARIO, CANADA: HUMAN FACTORS ASSOCIATION OF CANADA, 1978, 26-37.

DESCRIPTION:

THE PRODUCTION OF A HUMAN-ENGINEERED COMPUTER SYSTEM REQUIRES HUMAN FACTORS CONSIDERATIONS AT ALL STAGES OF DEVELOPMENT. THOUGH IT IS ESSENTIAL TO CONSIDER THE HUMAN ENGINEERING OF THE VISIBLE MAN-COMPUTER INTERFACE, MORE CONCERN MUST BE CONCENTRATED ON THE FACTORS INVOLVED IN THE CONSTRUCTION OF THAT INTERFACE. IN THE FACE OF A CONTINUING EXPLOSION IN SOFTWARE PRODUCTION, WE MUST CONCERN OURSELVES WITH THE INTERFACE BETWEEN THE PROGRAMMER AND HIS PROGRAM. THE PRODUCTION OF COST-EFFECTIVE HIGH QUALITY SOFTWARE REQUIRES AN UNDERSTANDING OF THE WORKERS PRODUCING IT, THEIR WORK ENVIRONMENT, AND THE TOOLS THEY USE. (A)

12P, 16R.

58 EFFECT OF SYSTEM RESPONSE TIME ON USER

CARBONELL, J.R., ELKIND, J.I., & NICKERSON, R.S. ON THE PSYCHOLOGICAL IMPORTANCE OF TIME IN A TIME SHARING SYSTEM. HUMAN FACTORS, 1968, 10, 135-142 (ALSO: REPORT NO. SCIENTIFIC-6, BBN-1687. CAMBRIDGE, MASSACHUSETTS: BOLT BERANEK AND NEWMAN, INC., SEPTEMBER 1967). (NTIS NO. AD 670604)

DESCRIPTION:

ONE OF THE MOST IMPORTANT PROBLEMS IN THE DESIGN AND/OR OPERATION OF A COMPUTER UTILITY IS TO OBTAIN DYNAMIC CHARACTERISTICS THAT ARE ACCEPTABLE AND CONVENIENT TO THE ON-LINE USER. THIS PAPER IS CONCERNED WITH THE PROBLEMS OF ACCESS TO THE COMPUTER UTILITY, RESPONSE TIME AND ITS EFFECT UPON CONVERSATIONAL USE OF THE COMPUTER, AND THE EFFECTS OF LOAD ON THE SYSTEM. PRIMARY ATTENTION IS PLACED UPON RESPONSE TIME: RATHER THAN A SINGLE MEASURE, A SET OF RESPONSE TIMES SHOULD BE MEASURED IN A GIVEN COMPUTER UTILITY, IN CORRESPONDENCE TO THE DIFFERENT TYPES OF OPERATIONS REQUESTED. IT IS ASSUMED THAT THE PSYCHOLOGICAL VALUE OF SHORT RESPONSE TIME STEMS FROM A SUBJECTIVE COST MEASURE OF THE USER'S OWN TIME, LARGELY INFLUENCED BY THE VALUE OF CONCURRENT TASKS BEING POSTPONED. A MEASURE OF COST (TO THE INDIVIDUAL AND/OR HIS ORGANIZATION) OF THE TIME-ON-LINE REQUIRED TO PERFORM A TASK MIGHT THUS BE DERIVED. MORE SUBTLE IS THE PROBLEM OF THE USER'S ACCEPTABILITY OF GIVEN RESPONSE TIMES. THIS ACCEPTABILITY IS A FUNCTION OF THE SERVICE REQUESTED (E.G., LENGTH OF COMPUTATION), AND VARIABILITY WITH RESPECT TO EXPECTATIONS DUE BOTH TO UNCERTAINTY IN THE USER'S ESTIMATION AND TO VARIATIONS IN THE RESPONSE TIME ORIGINATED BY VARIABLE LOADS ON THE SYSTEM. AN EFFORT SHOULD BE MADE BY COMPUTER-UTILITY DESIGNERS TO INCLUDE DYNAMIC CHARACTERISTICS (SUCH AS PREDICTION OF LOADS AND THEIR EFFECTS) AMONG THEIR DESIGN SPECIFICATIONS. (A)

8P, 6R.

59 DATA ENTRY ERRORS

CARLSON, G. PREDICTING CLERICAL ERROR IN AN EDP ENVIRONMENT. DATAMATION, FEBRUARY 19(2)63, 9, 34-36.

DESCRIPTION:

VERY LITTLE IS KNOWN ABOUT ERROR IN ANY PRECISE MANNER, EXCEPT THAT IT IS USUALLY PRESENT AND TROUBLESOME. THIS PAPER DESCRIBES AN ATTEMPT AT PREDICTING, OR SIMULATING, HUMAN ERROR. ERRORS MADE IN A NUMERIC DATA ENTRY TASK ARE ANALYZED AND A BINARY DECISION TREE IS DEVELOPED TO SIMULATE THESE ERRORS. THIS MODEL IS INDEPENDENT OF THE INDIVIDUAL OPERATOR AND TYPE OF WORK BEING DONE AND CORRECTLY PREDICTED 46% OF OBSERVED ERRORS. (MEA)

3P, 0R.

60 PROGRAMMING

CARSTENSEN, I., FISCHER, L., JORGENSEN, A.H., & WEISSMAN, L. AN EXPERIMENTAL STUDY OF PROGRAM READABILITY AND MODIFIABILITY. DATALOGISK INSTITUT, UNIVERSITY OF COPENHAGEN, 1975.

61 SPECIFIC QUERY LANGUAGE

CHAMBERLIN, D.D., & BOYCE, R.F. SEQUEL: A STRUCTURED ENGLISH QUERY LANGUAGE. IN PROCEEDINGS OF THE ACM SIGMOD WORKSHOP, MAY 1974, 242-264 (ALSO TECHNICAL REPORT RJ1394, IBM RESEARCH LABORATORY, SAN JOSE, CALIFORNIA, MAY 1974).

DESCRIPTION:

IN THIS PAPER, WE PRESENT THE DATA MANIPULATION FACILITY FOR A STRUCTURED ENGLISH QUERY LANGUAGE (SEQUEL) WHICH CAN BE USED FOR ACCESSING DATA IN AN INTEGRATED RELATIONAL DATA BASE. WITHOUT RESORTING TO THE CONCEPTS OF BOUND VARIABLES AND QUANTIFIERS SEQUEL IDENTIFIES A SET OF SIMPLE OPERATIONS ON TABULAR STRUCTURES, WHICH CAN BE SHOWN TO BE OF EQUIVALENT POWER TO THE FIRST ORDER PREDICATE CALCULUS. A SEQUEL USER IS PRESENTED WITH A CONSISTENT SET OF KEYWORD ENGLISH TEMPLATES WHICH REFLECT HOW PEOPLE USE TABLES TO OBTAIN INFORMATION. MOREOVER, THE SEQUEL USER IS ABLE TO COMPOSE THESE BASIC TEMPLATES IN A STRUCTURED MANNER IN ORDER TO FORM MORE COMPLEX QUERIES. SEQUEL IS INTENDED AS A DATA BASE SUBLANGUAGE FOR BOTH THE PROFESSIONAL PROGRAMMER AND THE MORE INFREQUENT DATA BASE USER. (A)

23P, 18R.

62 PROGRAMMER PRODUCTIVITY

CHAMPINE, G.A. ESTIMATING METHODS AND MEASURES OF PROGRAMMER PRODUCTIVITY (TECHNICAL REPORT). ROSEVILLE, MINNESOTA: SPERRY-UNIVAC, MAY 1973.

DESCRIPTION:

THE PURPOSE OF THIS PAPER IS TO REVIEW SEVERAL QUANTITATIVE MODELS FOR THE PROGRAMMING PROCESS. THE BASIS OF THE PARAMETERS IN THESE MODELS IS A RATHER CONSIDERABLE AMOUNT OF HISTORICAL WORK THAT HAS BEEN DONE IN THE AREA OF PROGRAMMING MANAGEMENT. THE PURPOSE OF THE MODELS IS TO PERMIT RELIABLE ESTIMATIONS TO BE MADE OF COST, SCHEDULE AND MANPOWER RESOURCES REQUIRED TO PERFORM A GIVEN PROGRAMMING JOB. A SECONDARY OBJECTIVE OF THE MODELS IS TO PROVIDE ADEQUATE STANDARDS OF PERFORMANCE BY WHICH PROJECT PERFORMANCE MAY BE COMPARED AFTER THE FACT. THE PARAMETERS OF THE JOB, PROGRAMMER AND WORKING ENVIRONMENT ARE REVIEWED AND THOSE WHICH CORRELATE WITH COST AND SCHEDULE ARE IDENTIFIED AND RELATED IN A QUALITATIVE MANNER. (A, ABBR.)

25P, 8R.

63 PROGRAMMER PRODUCTIVITY

CHAMPINE, G.A., & CARLSON, W.H. PROGRAMMER PRODUCTIVITY (TECHNICAL REPORT). ROSEVILLE, MINNESOTA: SPERRY-UNIVAC, UNDATED.

DESCRIPTION:

ONE OF THE KEY ISSUES IN MANAGEMENT OF SOFTWARE IS THE ESTIMATION, PREDICTION, AND CONTROL OF PROGRAMMING PRODUCTIVITY. THE TOPIC OF PROGRAMMING PRODUCTIVITY MAY BE DIVIDED INTO THE TOPICS OF

- IDENTIFICATION OF FACTORS AFFECTING PRODUCTIVITY
- ESTIMATION OF PRODUCTIVITY FACTORS
- PREDICTION OF PROGRAMMING COST AND SCHEDULE
- TECHNICAL METHODS OF IMPROVING PRODUCTIVITY
- MANAGEMENT TECHNIQUES FOR IMPROVING PRODUCTIVITY

EACH OF THESE TOPICS IS REVIEWED WITH RESPECT TO THE LITERATURE, AND CURRENT PRACTICE AS IT EXISTS IN INDUSTRY AND IN THE ROSEVILLE DEVELOPMENT CENTER.

A NUMBER OF STATISTICAL STUDIES HAVE BEEN DONE ON PROGRAMMING PRODUCTIVITY, BUT THEIR USE IN ACTUAL PRACTICE HAS BEEN VERY LIMITED. HOWEVER, PRODUCTIVITY HAS BEEN SUBSTANTIALLY IMPROVED THE LAST FEW YEARS THROUGH THE WIDESPREAD USE OF INTERACTIVE TERMINALS, STRUCTURED PROGRAMMING, IMPROVED DEBUGGING TOOLS, AND BETTER MANAGEMENT TECHNIQUES. UNFORTUNATELY, THE ENTIRE ISSUE IS CLOUDED BY THE LACK OF A SUITABLE STANDARD OF MEASURE. (A)

28P, 24R.

64 ROLLBACK AND RECOVERY STRATEGIES

CHANDY, K.M. A SURVEY OF ANALYTIC MODELS OF ROLLBACK AND RECOVERY STRATEGIES. COMPUTER MAGAZINE, MAY 1975, 8(5), 40-47.

DESCRIPTION:

THERE ARE SEVERAL WAYS TO INCREASE SYSTEM RELIABILITY. CHOOSING THE MOST COST-EFFECTIVE ONE IS NOT EASY, BUT MODELS SUCH AS THOSE OUTLINED IN THIS PAPER CAN HELP.

FOLLOWING A BRIEF DISCUSSION OF THE COST-EFFECTIVENESS OF REDUNDANCY SCHEMES, ATTENTION IS THEN RESTRICTED TO THE ANALYSIS OF ROLLBACK AND RECOVERY STRATEGIES. THE DIFFERENT OBJECTIVES AND CONSTRAINTS OF ROLLBACK AND RECOVERY STRATEGIES ARE DISCUSSED, USING A DATA-BASED SYSTEM AND A PROCESS-CONTROL SYSTEM AS EXAMPLES. APPROACHES TO MODELING THE ROLLBACK-RECOVERY PROCESS ARE PRESENTED, AND THE ANALYSIS OF THREE SPECIFIC MODELS IS REVIEWED. (O)

8P, 5R.

65 FLOWCHARTING

CHAPIN, N. FLOWCHARTING WITH THE ANSI STANDARD: A TUTORIAL. COMPUTING SURVEYS, 1970, 2, 119-146.

DESCRIPTION:

THE ISO AND ANSI X3.5 STANDARD FLOWCHART SYMBOLS AND THEIR USAGE IN INFORMATION PROCESSING ARE EXPLAINED AND EXAMPLES GIVEN. THE TWO MAIN CATEGORIES OF FLOWCHART -- THE SYSTEM CHART OR RUN DIAGRAM, AND THE FLOW DIAGRAM OR BLOCK DIAGRAM -- ARE STRESSED. FOR EACH, THE OUTLINE SYMBOLS AND THEIR MANNER OF USE ARE PRESENTED, AS WELL AS GUIDELINES AND CONVENTIONS, SUCH AS CROSS-REFERENCING. IN THE CASE OF FLOW DIAGRAMS, NOTATION IS PRESENTED FOR USE WITHIN THE OUTLINE SYMBOLS.

28P, 21R.

66 FLOWCHARTING

CHAPIN, N. NEW FORMAT FOR FLOWCHARTS. SOFTWARE PRACTICE AND EXPERIENCE, 1974, 4, 341-357.

DESCRIPTION:

THIS PAPER PROPOSES A NEW CHART FORMAT AS AN ALTERNATIVE FOR THE FLOW DIAGRAM VARIETY OF FLOWCHART. THIS NEW CHART FORMAT IS MORE INFORMATIVE, MORE COMPACT AND EASIER TO USE AND DRAW THAN THE ANS FORMAT, AND FACILITATES MODULARIZATION IN DESIGN AND PROGRAMMING. THE NEW CHART FORMAT SUPPORTS RESTRICTIONS ON CONTROL TRANSFERS, AND CLEARLY IDENTIFIES THE CONTROL STRUCTURES USED IN STRUCTURED PROGRAMMING. THE NEW CHART FORMAT ALSO PERMITS SHOWING MULTIPLE LEVELS OF DETAIL UNAMBIGUOUSLY IN A SINGLE FLOWCHART, AND ENABLES HIPO CHARTS AND SYSTEM CHARTS TO BE USEFULLY AUGMENTED. (A)

17P, 12R.

67 PROGRAM SYNTHESIS

CHATELIN, P. SELF-REDEFINITION AS A PROGRAM MANIPULATION STRATEGY. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 174-179 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 174-179).

DESCRIPTION:

THIS IS AN EXPLORATION OF A CONSTRUCTIVE STRATEGY FOR PROGRAM IMPROVEMENT AND SYNTHESIS. A FIRST PART RECALLS UNFOLDING-FOLDING STYLE OF MANIPULATIONS INITIATED BY BURSTALL AND DARLINGTON WITH AN APPLICATION TO PROOFS OF EQUIVALENCE OF CERTAIN FUNCTION COMPOSITIONS. SECOND PART, IN A MORE ABSTRACT WAY, PRESENTS THREE BASIC "FORMS" AND THEIR ASSOCIATED "TRANSFORMS" CONSTRUCTED WITH THIS STRATEGY IN A HIERARCHICAL ORDER; THEY MAY SERVE AS GOALS OF TRANSFORMATIONS. LAST PART ASSOCIATES SELF-REDEFINITION TO MIXED STRATEGIES FOR PROGRAM COMPOSITION: SYMBOLIC MACRO REPLACEMENT, LOGARITHMIC SPEED UP, RESOLUTION OF FORMAL RECURRENCES. EACH SITUATION, WHERE TECHNIQUE AND METHOD APPLY, IS DEPICTED ON EXAMPLES AND OPEN PROBLEMS ARE EVOKED. (A)

6P, 17R.

68 INTELLIGENT TERMINALS

CHEN, T.C. DISTRIBUTED INTELLIGENCE FOR USER-ORIENTED COMPUTING. AFIPS CONFERENCE PROCEEDINGS, 1972, 41, 1049-1056.

DESCRIPTION:

THE PRIMITIVES USED BY THE COMPUTER DESIGNER HAVE BLOSSOMED FROM THE SINGLE LOGICAL CONNECTIVES OF TWO DECADES AGO, INTO CHIPS CONTAINING THOUSANDS OF CIRCUITS AND BITS. YET, THE QUANTITATIVE ASPECT OF THE ACHIEVEMENT, IMPOSING AS IT IS, SIGNIFIES LESS THAN THE QUALITATIVE INJECTION OF MACHINE INTELLIGENCE DOWN TO THE CHIP LEVEL. WITH THE CONSEQUENT FREEDOM TO DISTRIBUTE COMPUTING POWER, MACHINE DESIGN ENTERS A NEW ERA.

WE ASSERT THAT VERY POWERFUL EXTENSIBLE SYSTEMS, BASED ON THE LOOSE-COUPLING OF NESTED AUTONOMOUS MODULES, CAN HARMONIZE WITH THE HARDWARE TRENDS AND BE DIRECTED TOWARD HUMAN-ORIENTED, INTERPRETIVE COMPUTING. THE KEY TO PERFORMANCE IS SELF-OPTIMIZATION CONDUCTED THROUGHOUT THE POLYCENTRIC SYSTEM. (A)

8P, 21R.

69 STRUCTURED PROGRAMMING

CHENG, L.L. ENGINEERING OF QUALITY SOFTWARE SYSTEMS (SOME CASE STUDIES IN STRUCTURED PROGRAMMING) (REPORT NO. MTR-2648-VOL-6). BEDFORD, MASSACHUSETTS: MITRE CORP., JANUARY 1975.

70 NATURAL LANGUAGE PROGRAMMING

CHODOROW, M.S., & MILLER, L.A. THE INTERPRETATION OF TEMPORAL ORDER IN COORDINATE CONJUNCTION (TECHNICAL REPORT NO. RC 6199). YORKTOWN HEIGHTS, NEW YORK: IBM THOMAS J. WATSON RESEARCH CENTER, 1976.

DESCRIPTION:

THIS PAPER PROVIDES A NON-CONTEXTUAL ANALYSIS OF THE TEMPORAL ORDER OF ACTIONS THAT ARE EXPRESSED AS COORDINATELY CONJOINED VERBS. ALTHOUGH THE ANALYSIS IS DEVELOPED TO ACCOUNT FOR PHENOMENA IN A PROCEDURE SPECIFICATION DOMAIN (THE COOKING RECIPE), ITS PRINCIPLES ARE BELIEVED TO BE GENERALLY APPLICABLE. COOKING INSTRUCTIONS CONTAINING A PAIR OF CONJOINED VERBS ARE INTERPRETED AS REQUIRING THE TWO ACTIONS TO BE PERFORMED EITHER CONSECUTIVELY OR SIMULTANEOUSLY. IF THE ACTIONS ARE COMPATIBLE, THEY MAY BE EXECUTED SIMULTANEOUSLY; IF THEY ARE INCOMPATIBLE, THEY MUST BE EXECUTED CONSECUTIVELY. COMPATIBILITY IS DEFINED IN TERMS OF PRECONDITIONS AND ON-GOING CONDITIONS FOR ACTIONS. CONSECUTIVE ACTIONS ARE OFTEN ACCOMPANIED BY INTERACTION EFFECTS WHICH CAN BE ATTRIBUTED TO PARTIALLY OR INCORRECTLY FULFILLED PRECONDITIONS. PRECONDITIONS AND COMPATIBILITY PROVIDE THE FRAMEWORK FOR A SUFFICIENT SOLUTION TO ONE TYPE OF INTERACTION PROBLEM. THE SET OF PRE- AND ON-GOING CONDITIONS FOR AN ACTION IS ENTAILED BY THE VERB WHICH EXPRESSES THAT ACTION. THIS ENTAILMENT RELATIONSHIP IS CONSISTENT WITH THE GENERAL REQUIREMENTS FOR A NON-CONTEXTUAL SOLUTION TO THE INTERPRETATION OF TEMPORAL ORDER. PREVIOUS LINGUISTIC APPROACHES CANNOT ACCOUNT FOR SIMULTANEOUS ACTIONS OR FOR INTERACTION PHENOMENA. (A)

71 DESIGN METHODOLOGY

CHU, Y. A METHODOLOGY FOR SOFTWARE ENGINEERING. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 262-270.

DESCRIPTION:

THIS PAPER PRESENTS A METHODOLOGY FOR SOFTWARE ENGINEERING. THIS METHODOLOGY RECOGNIZES THE EXISTENCE OF TWO SEPARATE AND DISTINCTIVE PHASES (ARCHITECTURE AND IMPLEMENTATION) OF A SOFTWARE ENGINEERING TASK. THESE TWO PHASES ARE INTERFACED BY A FORMALIZED BUT DESCRIPTIVE DESIGN SPECIFICATION DESCRIBED BY A LANGUAGE CALLED ADL (ARCHITECTURAL DESIGN LANGUAGE). THIS ADL DESCRIPTION WOULD SERVE A SIMILAR PURPOSE AS THAT SERVED BY THE BLUEPRINT. IMPLEMENTATION CAN THEN BE ACCOMPLISHED FROM THE "SOFTWARE BLUEPRINT" IN ANY OF THREE POSSIBILITIES: SOFTWARE, HARDWARE, OR MICROWARE. DESIGN OF A LEXICAL SCANNER IS CHOSEN AS AN EXAMPLE TO ILLUSTRATE THIS METHODOLOGY. (A)
9P.

72 SOFTWARE ENGINEERING

CLAPP, J.A. SOFTWARE ENGINEERING: PROBLEMS AND FUTURE DEVELOPMENTS (REPORT NO. MTR-2791). BEDFORD, MASSACHUSETTS: MITRE CORP., NOVEMBER 1974.

73 SOFTWARE DEVELOPMENT

CLAPP, J.A., & LAPADULA, L.J. ENGINEERING OF QUALITY SOFTWARE SYSTEMS (REPORT NO. MTR-2648-VOL-1). BEDFORD, MASSACHUSETTS: MITRE CORP., NOVEMBER 1974.

74 SOFTWARE DEVELOPMENT PROJECT MANAGEMENT

CLAPP, J.A., & SULLIVAN, J.E. AUTOMATED MONITORING OF SOFTWARE QUALITY. AFIPS CONFERENCE PROCEEDINGS, 1974, 43, 337-341.

DESCRIPTION:

ALTHOUGH SOFTWARE COSTS ARE FREQUENTLY DESCRIBED AS "HIGH," ALL WE REALLY KNOW FOR SURE IS THAT FOR LARGE SOFTWARE SYSTEMS WE DO NOT KNOW HOW TO ESTIMATE COST, THE RELATION BETWEEN COST AND QUALITY, OR HOW TO MEASURE QUALITY. THE PRINCIPAL PROBLEM IS THE LACK OF APPLICABLE MEASURES AND MEASURE-RELATING THEOREMS THAT WOULD BE USEFUL IN PLANNING AND CONTROLLING SOFTWARE DEVELOPMENT. THIS PAPER DESCRIBES SIMON, A PROPOSED AUTOMATED AID THAT EXTRACTS MEASURES THROUGHOUT THE SOFTWARE IMPLEMENTAYION EFFORT AND PROVIDES ANALYSES OF THE DATA TO MANAGERS AND PROGRAMMERS. (MEA)
5P, 6R.

75 PROGRAMMING LANGUAGES

CLARKE, K.E., & JOHNSON, C.B.C. A COMPARISON OF TIME-SHARING LANGUAGES. IN MAN-COMPUTER INTERACTION: PROCEEDINGS OF THE CONFERENCE ON MAN-COMPUTER INTERACTION, 2-4 SEPTEMBER 1970 (CONFERENCE PUBLICATION NO. 68). LONDON, ENGLAND: INSTITUTION OF ELECTRICAL ENGINEERS, 1970, 167-171.

DESCRIPTION:

THIS PAPER IS BASED ON THE AUTHORS' EXPERIENCE USING OVER A DOZEN TIME-SHARING SYSTEMS. IT IS CONCERNED WITH THE FACILITIES OFFERED TO THE USER RATHER THAN THE PROBLEMS OF IMPLEMENTATION. ALTHOUGH PRIMARILY CONCERNED WITH THE PROGRAMMING LANGUAGES IT INCLUDES A SECTION ON SYSTEM COMMANDS SINCE THESE AFFECT THE EASE WITH WHICH A SYSTEM MAY BE USED. THE PAPER IS BASED ON OPINIONS FORMED APPRAISING TIME-SHARING SYSTEMS FROM PROGRAMMING IN THE LANGUAGES AND FROM USE MADE OF THE SYSTEMS BY ENGINEERS. (A)

FORTRAN, ALGOL, COBOL, BASIC, JEAN, FIGARO, TELCOMP, DELTA, POP-2, APL, AND PL/1 ARE BRIEFLY EVALUATED WITH RESPECT TO POWER OF THE LANGUAGE, DEGREE OF INTERACTION, AND EASE OF LEARNING. (MEA)
5P, 0R.

76 SOFTWARE ENGINEERING

COMPUTER MAGAZINE, SPECIAL ISSUE ON SOFTWARE ENGINEERING, MAY 1975, 8(5).

77 STRUCTURED PROGRAMMING

COMPUTER MAGAZINE, SPECIAL ISSUE ON STRUCTURED PROGRAMMING, JUNE 1975, 8(5).

78 SOFTWARE DESIGN

CONWAY, M.E. HOW DO COMMITTEES INVENT? DATAMATION, APRIL 1968, 14(4), 28-31.

DESCRIPTION:

THE BASIC THESIS OF THIS PAPER IS THAT ORGANIZATIONS THAT DESIGN SYSTEMS WILL NECESSARILY PRODUCE DESIGNS THAT ARE STRUCTURALLY IDENTICAL TO THE COMMUNICATION STRUCTURES IN THESE ORGANIZATIONS. THE IMPLICATIONS OF THIS FACT FOR THE MANAGEMENT OF SYSTEM DESIGN ARE CONSIDERED, WITH PRIMARY EMPHASIS ON ORGANIZING THE DESIGN EFFORT ACCORDING TO THE NEED FOR COMMUNICATION. (MEA)
4P, 3R.

79 PROGRAMMING

COOKE, J.E., & BUNT, R.B. HUMAN ERROR IN PROGRAMMING: THE NEED TO STUDY THE INDIVIDUAL PROGRAMMER. INFOR, 1975, 13, 296-307 (ALSO: TECHNICAL REPORT NO. 75-3. SASKATOON, CANADA: UNIVERSITY OF SASKATOON, DEPARTMENT OF COMPUTATIONAL SCIENCE, 1975).

DESCRIPTION:

THIS PAPER DISCUSSES THE PROBLEM OF HUMAN ERROR IN PROGRAMMING, AND SUGGESTS THE IMPORTANCE OF EMPIRICAL VALIDATION OF THE EFFECTIVENESS OF NEW STYLES OR TECHNIQUES, SUCH AS THOSE GENERALLY CONSIDERED AS THE COMPONENTS OF "STRUCTURED PROGRAMMING." AS A COMPLEMENT TO STUDIES OF GROUPS OF PROGRAMMERS, IT IS SUGGESTED THAT STUDIES ARE NEEDED WHICH CONCENTRATE ON THE INFORMATION-PROCESSING LIMITATIONS OF HUMAN PROGRAMMERS.

FOLLOWING A REVIEW OF RELEVANT LITERATURE FROM OTHER DISCIPLINES, EYE-MOVEMENTS, PERCEPTION AND SHORT-TERM MEMORY ARE IDENTIFIED AS RELEVANT TO THE PROBLEM OF READING AND UNDERSTANDING A PROGRAM. A POSSIBLE EXPERIMENTAL APPROACH THAT MIGHT ANSWER A NUMBER OF CURRENT QUESTIONS ABOUT PROGRAM STRUCTURE IS SUGGESTED. (A)

80 PROGRAMMING

COOKE, L.H., JR. PROGRAMMING TIME VS. RUNNING TIME. DATAMATION, DECEMBER 1974, 20(12), 56-58.

DESCRIPTION:

A BRIEF EXPERIMENTAL STUDY IS REPORTED IN WHICH SIX PROGRAMMERS PROGRAMMED AND EXECUTED A PROCEDURE IN FOUR DIFFERENT PROGRAMMING LANGUAGES. HIGHER-ORDER INTERPRETIVE LANGUAGES REQUIRED LESS DEVELOPMENT TIME AND MORE EXECUTION TIME THAN DID LESS POWERFUL COMPILER LANGUAGES. THE TRADE-OFF BETWEEN DEVELOPMENT COST AND EXECUTION COST IS DISCUSSED. (HRR)
3P, 3R.

81 CHIEF PROGRAMMER TEAMS

COOKE, L.H., JR. THE CHIEF PROGRAMMER TEAM ADMINISTRATOR. DATAMATION, JUNE 1976, 22(6), 85-86.

DESCRIPTION:

THE CHIEF PROGRAMMER/LIBRARIAN TEAM CONCEPT HAS BEEN USED WIDELY AND WITH GOOD SUCCESS SINCE ITS INTRODUCTION. WITHIN DIFFERENT PROJECTS AND ORGANIZATIONS, THE ROLE OF THE LIBRARIAN IS FLEXIBLE AND GENERALLY CONSIDERED THE LEAST IMPORTANT JOB. ENRICHING THE ROLE OF THE LIBRARIAN NOT ONLY INSURES MORE PERSONAL SATISFACTION BUT ALSO INCREASES PROGRAMMING PRODUCTIVITY. THIS PAPER PROPOSES THAT THE TERM "PROJECT ADMINISTRATOR" REPLACES THAT OF "LIBRARIAN" AND THAT THIS INDIVIDUAL ASSUME SOME OF THE DUTIES FORMERLY ALLOCATED TO THE CHIEF PROGRAMMER. (MEA)
2P, 3R.

82 SOFTWARE MAINTENANCE
CORPORATION FOR INFORMATION SYSTEMS RESEARCH AND DEVELOPMENT/CIRAD. A STUDY OF FUNDAMENTAL FACTORS UNDERLYING SOFTWARE MAINTENANCE PROBLEMS (VOLS. 1 & 2) (REPORT NO. ESD-TR-72-121). L.G. HANSCOM FIELD, BEDFORD, MASSACHUSETTS: HQ ELECTRONIC SYSTEMS DIVISION (AFSC), DEPUTY FOR COMMAND AND MANAGEMENT SYSTEMS, 1971 (NTIS NOS. AD 739479 AND AD 739872)
DESCRIPTION:
"PROBLEMS FACED BY PROGRAMMERS WHO MUST MAINTAIN PROGRAMS SOMEONE ELSE WROTE" WERE IDENTIFIED. THEY WERE REDUCED TO THREE FUNDAMENTAL INHIBITING FACTORS: (1) THE LIMITED RATE AT WHICH PEOPLE CAN MAKE "RELEVANCE TESTS," (2) OVER-CONFIRMATION IN CLUES REQUIRED BEFORE HYPOTHESIS-TESTING, AND (3) HUMAN VULNERABILITY TO DISTRACTION AND PROCRASTINATION. STUDIES SUGGESTED COLLECTIVELY BY THESE FACTORS WERE CONDUCTED. THE STUDIES (1) ASCERTAINED THAT PROGRAMMERS TEND TO THINK IN TERMS OF CONCEPTUAL GROUPINGS WHOSE OBJECTIVE IDENTIFICATION WOULD BE HELPFUL, (2) INDICATED THAT IT WAS FEASIBLE TO TRACE THE PATH THE PROGRAMMER TAKES AS HE PREPARES TO MAKE A MODIFICATION, AND (3) IDENTIFIED A FEW TENTATIVE MEASURES OF THE DEGREE OF MAINTAINABILITY OF COMPUTER PROGRAMS. (A)
142P, 74R.

83 SYSTEM ANALYSIS TECHNIQUES
COUGER, J.D., & KNAPP, R.W. (EDS.) SYSTEM ANALYSIS TECHNIQUES. NEW YORK, NEW YORK: WILEY, 1974.

84 SOFTWARE RELIABILITY
CRANDON, L.H., & ANDERSON, P.G. COMPUTER PROGRAM RELIABILITY. COMPUTERS AND PEOPLE, JULY 1974, 23, PP. 18-22; 42.

85 SOFTWARE RELIABILITY
CULPEPPER, L.M. A SYSTEM FOR RELIABLE ENGINEERING SOFTWARE. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 174-178.
DESCRIPTION:
MUCH OF THE SOFTWARE DEVELOPED TO SUPPORT ENGINEERING DESIGN CALCULATIONS IS PRODUCED BY PERSONS WHOSE PRIMARY SKILL IS NOT THAT OF COMPUTER SCIENTIST. SOFTWARE VALIDATION TECHNIQUES SUITABLE FOR USE BY THESE ENGINEER-PROGRAMMERS HAVE BEEN UNDER INVESTIGATION BY THE NAVY AS PART OF A PROJECT DESIGNED TO INCREASE THE RELIABILITY, USABILITY, AND PORTABILITY OF ENGINEERING DESIGN SOFTWARE. THIS PAPER DESCRIBES THE GOALS AND RESULTS OF THE PROJECT AND DESCRIBES THE SOFTWARE VALIDATION PROCESS WHICH WAS DEVELOPED. A SOFTWARE VALIDATION TOOL WHICH WAS PRODUCED DURING THE PROJECT IS DESCRIBED AND COMPARED WITH SEVERAL OTHER TOOLS. SOME AREAS FOR FURTHER WORK ARE SUGGESTED. (A)
5P, 17R.

86 PROGRAMMING, GENERAL
D'AGAPAYEFF, A. PROGRAMMING: THE UNWANTED, UNLOVED PROFESSION. COMPUTERS AND PEOPLE, 1977, 26, 10-12.

87 MANAGEMENT OF SOFTWARE DEVELOPMENT
DALY, E.B. MANAGEMENT OF SOFTWARE DEVELOPMENT. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1977, SE-3, 230-242.

88 PROGRAMMING TOOLS

DAVIS, R. GENERALIZED PROCEDURE CALLING AND CONTENT-DIRECTED INVOCATION. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 45-54 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 45-54).

DESCRIPTION:

WE SUGGEST THAT THE CONCEPT OF A STRATEGY CAN PROFITABLY BE VIEWED AS KNOWLEDGE ABOUT HOW TO SELECT FROM AMONG A SET OF PLAUSIBLY USEFUL KNOWLEDGE SOURCES, AND EXPLORE THE FRAMEWORK FOR KNOWLEDGE ORGANIZATION WHICH THIS IMPLIES. WE DESCRIBE META RULES, A MEANS OF ENCODING STRATEGIES THAT HAS BEEN IMPLEMENTED IN A PROGRAM CALLED TEIRESIAS, AND EXPLORE THEIR UTILITY AND CONTRIBUTION TO PROBLEM SOLVING PERFORMANCE.

META RULES ARE ALSO CONSIDERED IN THE BROADER CONTEXT OF A TOOL FOR PROGRAMMING. WE SHOW THAT THEY CAN BE CONSIDERED A MEDIUM FOR EXPRESSING THE CRITERIA FOR RETRIEVAL OF KNOWLEDGE SOURCES IN A PROGRAM, AND HENCE, CAN BE USED TO DEFINE CONTROL REGIMES. THE UTILITY OF THIS AS A PROGRAMMING MECHANISM IS CONSIDERED.

FINALLY, WE DESCRIBE THE TECHNIQUE OF CONTENT-DIRECTED INVOCATION USED BY META RULES, AND CONSIDER ITS USE AS A WAY OF IMPLEMENTING STRATEGIES. IT IS ALSO CONSIDERED IN HISTORICAL PERSPECTIVE AS A KNOWLEDGE SOURCE INVOCATION TECHNIQUE, AND ITS ADVANTAGE OVER SOME EXISTING MECHANISMS LIKE GOAL-DIRECTED INVOCATION IS CONSIDERED. (A)

10P, 24R.

89 PROGRAMMING LANGUAGES

DE KLEER, J., DOYLE, J., STEELE, G.L., JR., & SUSSMAN, G.J. AMORD: EXPLICIT CONTROL OF REASONING. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 116-125 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 116-125).

DESCRIPTION:

THE CONSTRUCTION OF EXPERT PROBLEM-SOLVING SYSTEMS REQUIRES THE DEVELOPMENT OF TECHNIQUES FOR USING MODULAR REPRESENTATIONS OF KNOWLEDGE WITHOUT ENCOUNTERING COMBINATORIAL EXPLOSIONS IN THE SOLUTION EFFORT. THIS REPORT DESCRIBES AN APPROACH TO DEALING WITH THIS PROBLEM BASED ON MAKING SOME KNOWLEDGE WHICH IS USUALLY IMPLICITLY PART OF AN EXPERT PROBLEM SOLVER EXPLICIT, THUS ALLOWING THIS KNOWLEDGE ABOUT CONTROL TO BE MANIPULATED AND REASONED ABOUT. THE BASIC COMPONENTS OF THIS APPROACH INVOLVE USING EXPLICIT REPRESENTATIONS OF THE CONTROL STRUCTURE OF THE PROBLEM SOLVER, AND LINKING THIS AND OTHER KNOWLEDGE MANIPULATED BY THE EXPERT BY MEANS OF EXPLICIT DATA DEPENDENCIES.

10P, 34R.

90 PROGRAMMING

DEREMER, F., & KRON, H. PROGRAMMING-IN-THE-LARGE VS. PROGRAMMING-IN-THE-SMALL. IN PROCEEDINGS, INTERNATIONAL CONFERENCE ON RELIABLE SOFTWARE, 21-23 APRIL 1975, LOS ANGELES, CALIFORNIA (ALSO: SIGPLAN NOTICES, JUNE 1975, 10(6), 114-121).

DESCRIPTION:

THE ACTIVITY OF WRITING LARGE PROGRAMS IS DISTINGUISHED FROM THAT OF WRITING SMALL ONES. LARGE PROGRAMS ARE DEFINED AS SYSTEMS CONSISTING OF MANY SMALL PROGRAMS (MODULES), POSSIBLY WRITTEN BY DIFFERENT PEOPLE. LANGUAGES FOR PROGRAMMING-IN-THE-SMALL, I.E., LANGUAGES NOT UNLIKE THE COMMON PROGRAMMING LANGUAGES OF TODAY, ARE NEEDED FOR WRITING MODULES. ALSO NEEDED IS A "MODULE INTERCONNECTION LANGUAGE" FOR KNITTING THOSE MODULES TOGETHER INTO AN INTEGRATED WHOLE AND FOR PROVIDING AN OVERVIEW WHAT FORMALLY RECORDS THE INTENT OF THE PROGRAMMER(S) AND THAT CAN BE CHECKED FOR CONSISTENCY BY A COMPILER. THIS PAPER EXPLORES THE SOFTWARE RELIABILITY ASPECTS OF SUCH AN INTERCONNECTION LANGUAGE. EMPHASIS IS PLACED ON FACILITIES FOR INFORMATION HIDING AND FOR DEFINING LAYERS OF VIRTUAL MACHINES. (A)

8P, 20R.

91 PROGRAMMING

DIJKSTRA, E.W. PROGRAMMING CONSIDERED AS A HUMAN ACTIVITY. IN PROCEEDINGS OF THE IFIP, 1965, 1, 213-217.

DESCRIPTION:

THE PURPOSE OF THIS PAPER IS TO DEVELOP A BETTER UNDERSTANDING OF THE NATURE OF THE QUALITY OF PROGRAMS AND THE LANGUAGES IN WHICH THEY ARE EXPRESSED. THIS PAPER CONSIDERS THE EFFECTS OF BOTH CLARITY AND EFFICIENCY OF DIFFERENT PROGRAM STRUCTURES AND ALGORITHMIC LANGUAGE PROPERTIES. (MEA)
5P, 0R.

92 PROGRAMMING

DIJKSTRA, E.W. THE HUMBLE PROGRAMMER. COMMUNICATIONS OF THE ACM, 1972, 15, 859-866.

DESCRIPTION:

THIS ARTICLE PRESENTS A BRIEF REVIEW OF THE HISTORY OF PROGRAMMING VIA FIVE 'MILESTONE' PROJECTS -- MAINLY THE INTRODUCTION OF SEVERAL PROGRAMMING LANGUAGES. HARDWARE LIMITATIONS IN EARLY MACHINES RESULTED IN PROGRAMMERS RELYING ON A VARIETY OF 'CLEVER TRICKS.' A CHIEF CONSIDERATION OF EARLY PROGRAMMING WAS THE OPTIMIZATION OF COMPUTATIONAL EFFICIENCY. THE INTRODUCTION OF BETTER HARDWARE HAS ONLY EXPANDED THE USE OF CLEVER TRICKS TO THE POINT THAT WE NOW FACE A CRISIS. SIX ARGUMENTS ARE PRESENTEED TO ADVOCATE A REVERSAL IN THIS TREND AND LEAD TO A REVOLUTION IN SOFTWARE DEVELOPMENT. A 'HUMBLE PROGRAMMER' SHOULD APPROACH A TASK WITH FULL APPRECIATION OF ITS DIFFICULTY, USE MODEST AND ELEGANT PROGRAMMING LANGUAGES, AND RESPECT THE LIMITATIONS OF THE HUMAN MIND. (GDC)
6P, 6R.

93 PROGRAMMING

DIJKSTRA, E.W. A DISCIPLINE OF PROGRAMMING. ENGLEWOOD CLIFFS, NEW JERSEY: PRENTICE-HALL, 1976.

94 APPROPRIATE PROPERTIES OF TIME-SHARING SYSTEMS

DOHERTY, W.J., THOMPSON, C.H., & BOIES, S.J. AN ANALYSIS OF INTERACTIVE SYSTEM USAGE WITH RESPECT TO SOFTWARE, LINGUISTIC, AND SCHEDULING ATTRIBUTES. IN PROCEEDINGS OF THE 1972 INTERNATIONAL CONFERENCE ON CYBERNETICS AND SOCIETY. NEW YORK: INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC., 1972, 113-119 (ALSO TECHNICAL REPORT RC-3914, IBM WATSON RESEARCH CENTER, YORKTOWN HEIGHTS, NY, 1972).

DESCRIPTION:

A QUALITY INTERACTIVE SYSTEM SHOULD ENHANCE A PERSON'S ABILITY TO DO WORK IN A SATISFYING AND EFFICIENT MANNER. THIS PAPER IS ORGANIZED AROUND THREE ASPECTS OF QUALITY FOR AN INTERACTIVE SYSTEM. WE WILL PRESENT OUR EXPERIENCE WITH RESPECT TO (1) THE FUNCTIONS (SOFTWARE) WHICH SHOULD BE INCLUDED IN AN INTERACTIVE SYSTEM, (2) THE TECHNIQUES REQUIRED TO ACHIEVE AND MAINTAIN USER SATISFACTION AND (3) THE TECHNIQUES NECESSARY FOR EFFECTIVE INSTALLATION MANAGEMENT. THE IMPORTANCE OF EDITING, THE PROBLEMS OF ON-LINE DATA MANAGEMENT, AND RESPONSE TIME CORRELATIONS TO USER BEHAVIOR ARE SOME OF THE KEY FINDINGS PRESENTED HERE. OUR OBSERVATION THAT INTERACTIVE SYSTEMS ARE EVOLUTIONARY IN NATURE, CONTINUALLY REFLECTING THE GROWTH OF NEW APPLICATIONS, NEW HARDWARE, AND NEW UNDERSTANDING OF WHAT IS POSSIBLE, IMPLIES THAT PROVISIONS MUST BE MADE FOR CONTROLLING AND NOURISHING THIS GROWTH IN A STIMULATING AND RESPONSIVE WAY. (A)
7P, 10R.

95 STRUCTURED PROGRAMMING

DONALDSON, J.R. STRUCTURED PROGRAMMING. DATAMATION, DECEMBER 1973, 19(1), 52-54.

DESCRIPTION:

RECENT SHIFTS IN EMPHASIS HAVE OCCURRED IN THE FIELD OF SOFTWARE DEVELOPMENT. THE PRIMARY REQUIREMENT TO BE MET IN SOFTWARE DEVELOPMENT HAS ALWAYS BEEN TO PERFORM THE FUNCTION SPECIFIED FOR THE SOFTWARE. BUT, WHERE AT ONE TIME SECONDARY EMPHASIS WAS PLACED ONLY ON SOFTWARE EFFICIENCY, TODAY THREE OTHER FACTORS ARE RECOGNIZED AS REQUIRING SPECIAL EMPHASIS -- RELIABILITY, MAINTAINABILITY, AND EXTENSIBILITY. SOFTWARE MAINTENANCE AND MODIFICATION ACCOUNT FOR A SUBSTANTIAL PORTION OF SOFTWARE EXPENDITURES. THIS CAN BE COUNTERACTED BY DESIGNING AND IMPLEMENTING SOFTWARE IN A WAY THAT MINIMIZES ERRORS AND MAXIMIZES MODIFIABILITY. (A, ABBR.)

3P, OR.

96 DATA ORGANIZATION

DURDING, B.M., BECKER, C.A., & GOULD, J.D. DATA ORGANIZATION. HUMAN FACTORS, 1977, 19, 1-14.

DESCRIPTION:

THREE EXPERIMENTS INVESTIGATED HOW PEOPLE ORGANIZE DATA. SUBJECTS WERE GIVEN SETS OF 15-20 WORDS AND ASKED TO ORGANIZE THEM ON PAPER. EACH WORD SET HAD A PRE-DEFINED ORGANIZATION (HIERARCHY, NETWORK, LISTS, TABLE) BASED ON THE SEMANTIC RELATIONS AMONG THE WORDS. EXPERIMENT 1 SHOWED THAT COLLEGE STUDENTS HAVE ALL THESE ORGANIZATIONAL STRUCTURES AVAILABLE FOR USE. THEY ORGANIZED MOST WORD SETS ON THE BASIS OF THE SEMANTIC RELATIONS INHERENT IN THEM. WHEREAS MOST SUBJECTS USED "APPROPRIATE" ORGANIZATIONS (THOSE THAT MOST EASILY PRESERVED THE RELATIONS), A FEW SUBJECTS ORGANIZED NEARLY ALL WORD SETS INTO LISTS. EXPERIMENT 2 SHOWED THAT SUBJECTS CAN EFFICIENTLY FIT THE WORD SETS INTO "SKELETONS" THAT WERE EXPLICITLY DESIGNED TO MAINTAIN ALL THE SEMANTIC RELATIONS AMONG THE WORDS. EXPERIMENT 3 SHOWED THAT SUBJECTS HAVE DIFFICULTY IN PRESERVING THE RELATIONS AMONG THE WORDS WHEN THEY WERE REQUIRED TO ORGANIZE THEM INTO INAPPROPRIATE STRUCTURES. THESE RESULTS ARE EVALUATED RELATIVE TO THE USE OF COMPUTER-BASED INFORMATION RETRIEVAL SYSTEMS. (A)

97 PROGRAMMING LANGUAGES

ELSHOFF, J.L. AN ANALYSIS OF SOME COMMERCIAL PL/1 PROGRAMS. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1976, SE-2, 113-120.

98 SOFTWARE PHYSICS

ELSHOFF, J.L. MEASURING COMMERCIAL PL/I PROGRAMS USING HALSTEAD'S CRITERIA. SIGPLAN NOTICES, MAY 1976, 11(5), 38-46.

DESCRIPTION:

THE CORRELATION BETWEEN NO (OBSERVED LENGTH) AND NC (CALCULATED LENGTH BASED OF HALSTEAD'S "MENTAL EFFORT" METRIC) OF 120 NON-STRUCTURED PROGRAMS WAS FOUND TO BE .976. CORRELATION BETWEEN NO AND NC FOR 32 STRUCTURED PROGRAMS WAS .985. FOR NON-STRUCTURED PROGRAMS, THE AVERAGE DIFFERENCE BETWEEN NO AND NC WAS -186; FOR STRUCTURED PROGRAMS, THIS VALUE WAS 215. THIS IMPLIES THAT IT IS POSSIBLE TO PREDICT THE STRUCTURE OF A PROGRAM BY THE DIRECTION AND MAGNITUDE OF THE DIFFERENCE BETWEEN NO AND NC. (O)

9P, 1CR.

99 SOFTWARE METRICS

ELSHOFF, J.L. A NUMERICAL PROFILE OF COMMERCIAL PL/1 PROGRAMS. SOFTWARE: PRACTICE AND EXPERIENCE, 1976, 6, 505-525.

DESCRIPTION:

A SAMPLE OF 120 PRODUCTION PL/1 PROGRAMS FROM SEVERAL COMMERCIAL COMPUTING INSTALLATIONS HAS BEEN STUDIED. DATA ABOUT THE PROGRAMS IN THE SAMPLE HAS BEEN EXTRACTED BY A PL/1 SCANNING PROGRAM. THE STATISTICAL RESULTS OF THE STUDY ARE PRESENTED IN THIS DOCUMENT.

THE PAPER CONCENTRATES ON STATISTICAL DATA AND NOT ON GENERAL CONCLUSIONS. THE DATA ARE ONLY INTERPRETED TO THE EXTENT THAT IT IS NOT ILL-DEFINED AND MISLEADING. THE DATA PROFILE THE USE OF BASIC PL/1 ELEMENTS AND THE STRUCTURE OF PROGRAMS WRITTEN IN PL/1. THE READER OF THIS REPORT WILL GET A BETTER UNDERSTANDING OF HOW PL/1 HAS BEEN USED IN THE COMMERCIAL ENVIRONMENT UP TO 1974. (A)

21P, 5R.

100 STRUCTURED PROGRAMMING

ELSHOFF, J.L. THE INFLUENCE OF STRUCTURED PROGRAMMING ON PL/1 PROGRAM PROFILES (RESEARCH PUBLICATION NO. GMR-2011). WARREN, MICHIGAN: GENERAL MOTORS CORP., JANUARY 1976.

DESCRIPTION:

TWO SETS OF COMMERCIAL PL/1 PROGRAMS ARE STUDIED. THE SETS REPRESENT PROGRAMMING PRACTICE BEFORE AND AFTER THE INTRODUCTION OF STRUCTURED PROGRAMMING TECHNIQUES. THE USE OF STRUCTURED PROGRAMMING IS FOUND TO MAKE A MEASURABLE DIFFERENCE IN THE QUALITY OF PROGRAMS. A FEW MINOR CHANGES IN THE USE OF THE PL/1 PROGRAMMING LANGUAGE ARE NOTED. SUBSTANTIAL MODIFICATIONS TO THE CONTROL STRUCTURE OF THE PROGRAMS ARE MEASURED. ALSO, SOME IMPROVEMENTS IN THE QUALITATIVE ASPECTS OF THE PROGRAMS ARE DISCUSSED. ALTHOUGH THE PROGRAMS ARE MUCH IMPROVED, FURTHER ALTERATIONS CAN MAKE THE PROGRAMS STILL BETTER. THE AUTHOR CONCLUDES THAT THE TIME AND TRAINING REQUIRED TO INTRODUCE STRUCTURED PROGRAMMING TECHNIQUES TO PROGRAMMERS WILL BEGIN PAYING DIVIDENDS IN THE FORM OF BETTER PROGRAMS WITHIN SIX MONTHS. (A)

17P, 5R.

101 DEBUGGING

ENABIT, R.S. A NEW APPROACH TO ON-LINE, RUN-TIME PROGRAM LOGIC AND ERROR DEBUGGING USING HARDWARE IMPLEMENTATION. BEHAVIORAL RESEARCH METHODS & INSTRUMENTATION, 1970, 2, 33-37.

DESCRIPTION:

WHAT IS BELIEVED TO BE A SOMEWHAT DIFFERENT APPROACH TO RAPID PROGRAM DEBUGGING HAS BEEN DEVISED IN WHICH: (1) EXECUTION OF THE PROGRAMMER'S LOGICAL THINKING IS AUTOMATICALLY DEBUGGED BY THE COMPUTER AT RUN TIME AND (2) STATUS ERRORS OF ANY TYPE MAY BE FED BACK INTO THE COMPUTER, WHICH SUBSEQUENTLY OUTPUTS THE IMMEDIATE STEPS LEADING TO THAT ERROR. DEBUGGING IS CARRIED OUT BY THE COMPUTER ON AN INSTRUCTION-BY-INSTRUCTION BASIS, ON-LINE, WITH ALL OR SELECTED INTERRUPTS SERVICED. A HARDWARE-SOFTWARE IMPLEMENTATION PACKAGE FOR THE PDP-8 IS DESCRIBED, WHICH COULD BE ADAPTED TO OTHER COMPUTERS AS WELL. (A)

102 ERRORS
ENDRES, A. AN ANALYSIS OF ERRORS AND THEIR CAUSES IN SYSTEM PROGRAMS. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 140-149 (ALSO IN PROCEEDINGS, INTERNATIONAL CONFERENCE ON RELIABLE SOFTWARE, 21-23 APRIL 1975. SIGPLAN NOTICES, JUNE 1975, 10(6), 327-336).
DESCRIPTION:
PROGRAM ERRORS DETECTED DURING INTERNAL TESTING OF THE OPERATING SYSTEM DOS/VS FORM THE BASIS FOR AN INVESTIGATION OF ERROR DISTRIBUTIONS IN SYSTEM PROGRAMS. USING A CLASSIFICATION OF THE ERRORS ACCORDING TO VARIOUS ATTRIBUTES, CONCLUSIONS CAN BE DRAWN CONCERNING THE POSSIBLE CAUSES OF THESE ERRORS. THE INFORMATION THUS OBTAINED IS APPLIED IN A DISCUSSION OF THE MOST EFFECTIVE METHODS FOR THE DETECTION AND PREVENTION OF ERRORS. (A)
10P, 8R.

103 SOFTWARE DEVELOPMENT
ENGLEMAN, C. ENGINEERING OF QUALITY SOFTWARE SYSTEMS (TOWARD AN ANALYSIS OF THE LISP PROGRAMMING LANGUAGE) (REPORT NUMBER MTR-2648-VOL-4). BEDFORD, MASSACHUSETTS: MITRE CORP., JANUARY 1975.

104 DEBUGGING
ERICKSON, W.J. A PILOT STUDY OF INTERACTIVE VERSUS NONINTERACTIVE DEBUGGING (SDC TECHNICAL REPORT NO. TM-3296). SANTA MONICA, CALIFORNIA: SYSTEM DEVELOPMENT CORP., DECEMBER 1966.

105 MEASUREMENT OF TIME-SHARING PERFORMANCE
EVANS, R.C., & MILLER, L.A. STARCAT: A SYSTEM TO ANALYZE INTERACTIVE CMS PERFORMANCE (TECHNICAL REPORT RC-7072). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, APRIL 1978.

106 INTERACTIVE DEBUGGING
EVANS, T.G., & DARLEY, D.L. ON-LINE DEBUGGING TECHNIQUES: A SURVEY. AFIPS CONFERENCE PROCEEDINGS, 1966, 29, 37-50.
DESCRIPTION:
THIS PAPER IS A SURVEY OF ON-LINE DEBUGGING TECHNIQUES. FIRST, WE INTRODUCE THE READER WHO IS UNFAMILIAR WITH ON-LINE DEBUGGING TO THE CAPABILITIES OF CURRENTLY AVAILABLE SYSTEMS AND THEN CONSIDER IN SOME DETAIL THE MOST IMPORTANT FEATURES OF THESE SYSTEMS. SECOND, WE WILL CONSIDER THE PRINCIPAL FEATURES OF PAST AND PRESENT ON-LINE DEBUGGING SYSTEMS, TOGETHER WITH REMARKS ON IMPLEMENTATION, DISPLAYS, COMPILER CONSTRUCTION, AND HARDWARE. ANNOTATED EXAMPLES OF CURRENT ON-LINE DEBUGGING METHODS ARE PRESENTED FOR ILLUSTRATION. (MEA)
14P, 28R.

107 PROGRAMMING LANGUAGES
EVERSHED, D.G., & RIPPON, G.E. HIGH LEVEL LANGUAGES FOR LOW LEVEL USERS. COMPUTER JOURNAL, 1971, 14, 87-90.
DESCRIPTION:
DESPITE THE PRESENCE OF "HIGH LEVEL" LANGUAGES, A COMMUNICATION BARRIER STILL EXISTS BETWEEN THE MAJORITY OF PEOPLE AND COMPUTERS. THIS PAPER SUGGESTS HOW SOME PRESENT COMPUTER LANGUAGES MAY BE IMPROVED, AND ATTEMPTS TO JUSTIFY THE APPLICATION OF INCREASED EFFORT TO THIS SUBJECT. (A)
FOUR AREAS THAT COULD LEAD TO REDUCING THE ERRORS INDUCED BY PRESENT HIGH LEVEL LANGUAGES ARE CONSIDERED. THESE AREAS ARE I/O ROUTINES, GENERAL COMPUTING INSTRUCTIONS, ERROR REPORTING, AND A CONVERSATIONAL MODE OF USE. THE INCREASING NUMBER OF OCCASIONAL COMPUTER USERS PROVIDES ECONOMIC JUSTIFICATION FOR IMPROVING PROGRAMMING LANGUAGES. (MEA)
4P, 0R.

108 PROGRAMMING
FAGAN, M. DESIGN AND CODE INSPECTIONS. IBM SYSTEMS JOURNAL, 1976, 15(3), 182-211.
DESCRIPTION:
SUBSTANTIAL NET IMPROVEMENTS IN PROGRAMMING QUALITY AND PRODUCTIVITY HAVE BEEN OBTAINED THROUGH THE USE OF FORMAL INSPECTIONS OF DESIGN AND OF CODE. IMPROVEMENTS ARE MADE POSSIBLE BY A SYSTEMATIC AND EFFICIENT DESIGN AND CODE VERIFICATION PROCESS WITH WELL-DEFINED ROLES FOR INSPECTION PARTICIPANTS. THE MANNER IN WHICH INSPECTION DATA IS CATEGORIED AND MADE SUITABLE FOR PROCESS ANALYSIS IS AN IMPORTANT FACTOR IN ATTAINING THE IMPROVEMENTS. IT IS SHOWN THAT BY USING INSPECTION RESULTS, A MECHANISM FOR INITIAL ERROR REDUCTION FOLLOWED BY EVER-IMPROVING ERROR RATES CAN BE ACHIEVED. (A)
30P, 9R.

109 PROGRAM TESTING
FAGAN, M.E. INSPECTING SOFTWARE DESIGN AND CODE. DATAMATION, OCTOBER 1977, 23(10), PP. 133-135; 138; 142-144.
DESCRIPTION:
SUCCESSFUL MANAGEMENT OF ANY PROCESS REQUIRES PLANNING, MEASUREMENT, AND CONTROL. IN PROGRAM DEVELOPMENT, THESE REQUIREMENTS TRANSLATE INTO DEFINING THE PROGRAMMING PROCESS IN TERMS OF A SERIES OF OPERATIONS, EACH HAVING ITS OWN EXIT CRITERIA. NEXT THERE MUST BE SOME MEANS OF MEASURING COMPLETENESS OF THE PRODUCT AT ANY POINT OF ITS DEVELOPMENT BY INSPECTIONS AND TESTING. AND FINALLY, THE MEASURED DATA MUST BE USED FOR CONTROLLING THE PROCESS.
DESIGN AND CODE INSPECTIONS HAVE BEEN APPLIED SUCCESSFULLY IN SEVERAL PROGRAMMING PROJECTS, BOTH LARGE AND SMALL, AND INCLUDING SYSTEMS AND APPLICATIONS PROGRAMS. THEY HAVE NOT BEEN FOUND TO "GET IN THE WAY" OF PROGRAMMING, BUT INSTEAD ENABLED HIGHER PREDICTABILITY THAN OTHER MEANS AND IMPROVED PRODUCTIVITY AND PRODUCT QUALITY. (A)
6P, 0R.

110 SOFTWARE ENGINEERING
FAIRLEY, R.E. AN EXPERIMENTAL PROGRAM-TESTING FACILITY. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 350-357.

111 SOFTWARE PHYSICS

FITZSIMMONS, A., & LOVE, L.T. A REVIEW AND CRITIQUE OF HALSTEAD'S THEORY OF SOFTWARE PHYSICS (TECHNICAL REPORT NO. 76ISP004). ARLINGTON, VIRGINIA: GENERAL ELECTRIC COMPANY, 1976.

DESCRIPTION:

DURING RECENT YEARS, THERE HAVE BEEN AN INCREASING NUMBER OF ATTEMPTS TO DEFINE AND TO MEASURE THE "COMPLEXITY" OF A COMPUTER PROGRAM. HALSTEAD HAS DEVELOPED A THEORY WHICH NOT ONLY PROVIDES A PRECISE OBJECTIVE MEASURE OF THE COMPLEXITY OF EXISTING SOFTWARE, BUT IT ALSO INCLUDES A METHOD FOR COMPUTING THE AMOUNT OF TIME REQUIRED TO IMPLEMENT A GIVEN PROGRAM. THIS PAPER PRESENTS HALSTEAD'S THEORY, KNOWN AS SOFTWARE PHYSICS, AND REVIEWS AND CRITIQUES THE MAJOR STUDIES AND EXPERIMENTS RELATING TO IT. THE AUTHORS HAVE PERFORMED SOME EX POST FACTO TESTS OF THE THEORY AND HAVE SUBSEQUENTLY DEVELOPED REFINEMENTS TO THE THEORY. THESE VERIFICATIONS AND REFINEMENTS ARE ALSO PRESENTED IN THIS PAPER. (A)

22P, 26R.

112 SOFTWARE PHYSICS

FITZSIMMONS, A., & LOVE, T. A REVIEW AND EVALUATION OF SOFTWARE SCIENCE. COMPUTING SURVEYS, MARCH 1978, 10(1), 3-18.

DESCRIPTION:

DURING RECENT YEARS, THERE HAVE BEEN MANY ATTEMPTS TO DEFINE AND MEASURE THE "COMPLEXITY" OF A COMPUTER PROGRAM. MAURICE HALSTEAD HAS DEVELOPED A THEORY THAT GIVES OBJECTIVE MEASURES OF SOFTWARE COMPLEXITY. VARIOUS STUDIES AND EXPERIMENTS HAVE SHOWN THAT THE THEORY'S PREDICTIONS OF THE NUMBER OF BUGS IN PROGRAMS AND OF THE TIME REQUIRED TO IMPLEMENT A PROGRAM ARE AMAZINGLY ACCURATE. IT IS A PROMISING THEORY WORTHY OF MUCH MORE PROBING SCIENTIFIC INVESTIGATION.

THIS PAPER REVIEWS THE THEORY, CALLED "SOFTWARE SCIENCE," AND THE EVIDENCE SUPPORTING IT. (A)

16P, 41R.

113 COMPUTER-ASSISTED INSTRUCTION

FRANCIS, L. THE TUTOR TRAINING COURSE: LESSONS LEARNED. URBANA-CHAMPAIGN, ILLINOIS: UNIVERSITY OF ILLINOIS, COMPUTER-BASED EDUCATION RESEARCH LABORATORY, 1976.

DESCRIPTION:

THE MILITARY TRAINING CENTERS (MTC) GROUP CREATED AND TAUGHT THE FIRST FORMAL AUTHOR TRAINING COURSE FOR THE TUTOR PROGRAMMING LANGUAGE AND THE USE OF THE PLATO SYSTEM. THE COURSE WAS USED OVER A PERIOD OF THREE YEARS TO TRAIN APPROXIMATELY 100 AUTHORS WHO SPENT TWO TO THREE WEEKS AT THE COMPUTER-BASED EDUCATION RESEARCH LABORATORY (CERL) OF THE UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN. IN GENERAL, THE NEW AUTHORS HAD LITTLE PREVIOUS EXPERIENCE WITH COMPUTERS OR PROGRAMMED INSTRUCTION. HOWEVER, MANY WERE CLASSROOM INSTRUCTORS.

THIRTEEN COGNITIVE AND AFFECTIVE PRINCIPLES GUIDED THE CREATION OF THE AUTHOR COURSE; FIVE WERE POSITIED ORIGINALLY, AND THE REST WERE FORMULATED BASED ON EXPERIENCE DERIVED FROM TEACHING THE COURSE. THIS REPORT CONTAINS A STATEMENT OF THE PRINCIPLES AND A DESCRIPTION OF THEIR IMPLEMENTATION, INCLUDING MANY EXAMPLES FROM COURSE MATERIALS. IT ALSO RECOUNTS THE HIGHLIGHTS AND TURNING POINTS OF THE AUTHOR TRAINING COURSE, REVIEWS THE BASIS FOR ITS MODIFICATION, AND EXAMINES THE DILEMMAS ENCOUNTERED IN TEACHING NEW AUTHORS TO PREPARE COMPUTER-BASED INSTRUCTIONAL MATERIALS. TECHNIQUES FOR RESOLVING SOME OF THESE DILEMMAS ARE SUGGESTED.

THE MTC TRAINING COURSE WAS EVALUATED BY EXAMINING THE EXPERIENCE OF USERS OUTSIDE OF MTC. THE OPINIONS AND RECOMMENDATIONS OF THESE OUTSIDE GROUPS SUGGEST THAT THEY FOUND THE MATERIALS VALUABLE AND EFFECTIVE. THIS REPORT IS DIRECTED TO INSTRUCTORS OF NEW AUTHORS, DEVELOPERS OF AUTHOR TRAINING MATERIALS, AND MANAGERS OF COMPUTER-BASED INSTRUCTION DEVELOPMENT CENTERS. (A)

114 AUTOMATIC PROGRAMMING

FRASER, C.W. A KNOWLEDGE-BASED CODE GENERATOR GENERATOR. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 126-129 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 126-129).

DESCRIPTION:

XGEN IS A PROGRAM THAT ACCEPTS A MACHINE DESCRIPTION AND PRODUCES A GOOD LOCAL CODE GENERATOR FOR AN ALGOL-LIKE LANGUAGE. IT IS ORGANIZED AS A PRODUCTION SYSTEM OF RULES CODIFYING PREVIOUSLY ACQUIRED HUMAN SKILLS FOR DEALING WITH COMPUTER ARCHITECTURE AND PROGRAMMING LANGUAGES. (A)
4P, 15R.

115 SOFTWARE DESIGN

FREEMAN, P. TOWARD IMPROVED REVIEW OF SOFTWARE DESIGNS. AFIPS CONFERENCE PROCEEDINGS, 1975, 44, 329-334.

DESCRIPTION:

DEVELOPMENT OF TECHNIQUES FOR THE REVIEW OF SOFTWARE DESIGNS HAS BEEN LARGELY NEGLECTED. THIS PAPER DESCRIBES A METHODOLOGY, DESIGN RATIONALIZATION THAT IS INTENDED TO IMPROVE THE REVIEWABILITY OF DESIGNS. THE PRINCIPAL COMPONENT OF DESIGN RATIONALIZATION IS THE EXPLICIT RECORDING OF THE ALTERNATIVES AND EVALUATIONS OF THESE ALTERNATIVES AT EACH DECISION POINT IN THE DESIGN. THIS INFORMATION, THAT PROVIDES THE REASONS BEHIND FEATURES IN THE DESIGN, PERMITS INDEPENDENT REVIEW OF DESIGN CHOICES AND THE REASONING UNDERLYING THESE CHOICES. (MEA)
6P, 11R.

116 SOFTWARE DESIGN TECHNIQUES, COLLECTED READINGS

FREEMAN, P., & WASSERMAN, A.I. (EDS.) TUTORIAL ON SOFTWARE DESIGN TECHNIQUES (2ND ED.). LONG BEACH, CALIFORNIA: IEEE COMPUTER SOCIETY, 1977.

117 PROGRAMMING

FRIEDENTHAL, T.H. MAINTENANCE/PROGRAMMING PANEL. STAMFORD, CONNECTICUT: STELMA INC., OCTOBER 1972. (NTIS NO. AD 905296L)

118 SPECIFIC PROGRAMMING LANGUAGE

FRIEND, J.E. SUPPLEMENTARY HANDBOOK FOR INTRODUCTION TO AID PROGRAMMING. STANFORD, CALIFORNIA: STANFORD UNIVERSITY, INSTITUTE FOR MATHEMATICAL STUDIES IN THE SOCIAL SCIENCES, 1972.

119 COMPUTER-ASSISTED INSTRUCTION

FRIEND, J.E. COMPUTER-ASSISTED INSTRUCTION IN PROGRAMMING: A CURRICULUM DESCRIPTION (TECHNICAL REPORT NO. 211). STANFORD, CALIFORNIA: STANFORD UNIVERSITY, INSTITUTE FOR MATHEMATICAL STUDIES IN SOCIAL SCIENCES, JULY 1973. (A)

DESCRIPTION:

THE COURSE PROVIDES AN INTRODUCTION TO COMPUTER PROGRAMMING FOR COMMUNITY COLLEGE STUDENTS WHO HAVE TAKEN HIGH SCHOOL ALGEBRA, AND IT IS EQUIVALENT TO A THREE QUARTER-UNIT COURSE IN COMPUTER SCIENCE. ALL INSTRUCTION IS PRESENTED BY COMPUTER, AND A SUPPLEMENTARY STUDENT MANUAL IS PROVIDED FOR REFERENCE. THE COURSE CONTENT RESEMBLES THAT OF OTHER INTRODUCTORY COURSES IN COMPUTER PROGRAMMING AND INCLUDES THE TOPICS OF STORED PROGRAMS, USE OF FUNCTIONS AND SUBROUTINES, CONDITIONAL CLAUSES, AND BRANCHING TECHNIQUES. THE INSTRUCTIONAL SYSTEM IMPLEMENTS UNDER COMPUTER CONTROL TEACHING STRATEGIES THAT MIGHT BE USED BY A HUMAN TUTOR SUCH AS INDIVIDUALIZING THE CONTENT, PACE, AND SEQUENCE OF INSTRUCTION, ALLOWING FOR SUFFICIENT STUDENT CONTROL, TAILORING WRONG ANSWER MESSAGES, AND PROVIDING BOTH REMEDIAL AND EXTRA-CREDIT WORK. STUDENTS ARE REQUIRED TO INTERACT ON-LINE WITH A COMMERCIALLY PREPARED EDITOR-INTERPRETER SIMILAR TO THOSE FOUND IN MANY TIME-SHARING ENVIRONMENTS.

PERFORMANCE DATA FROM BOTH THE INSTRUCTIONAL PROGRAM AND THE EDITOR-INTERPRETER ARE AUTOMATICALLY STORED FOR LATER RETRIEVAL AND ANALYSIS. THE ORGANIZATION OF THE COURSE, TYPES OF EXERCISES USED, AND CONTENT OF EACH LESSON ARE DOCUMENTED AND AN APPENDIX LISTS THE CONCEPTS ASSOCIATED WITH EACH EXERCISE IN THE COURSE. (A)

120 PROGRAMMING

FRIEND, J.E. 100 PROGRAMMING PROBLEMS. STANFORD, CALIFORNIA: STANFORD UNIVERSITY, INSTITUTE FOR MATHEMATICAL STUDIES IN THE SOCIAL SCIENCES, 1973. (B)

121 COMPUTER-ASSISTED INSTRUCTION

FRIEND, J. PROGRAMS STUDENTS WRITE (TECHNICAL REPORT NO. 257). STANFORD, CALIFORNIA: STANFORD UNIVERSITY, INSTITUTE FOR MATHEMATICAL STUDIES IN THE SOCIAL SCIENCES, JULY 1975. (NTIS NO. AD AC15093)

DESCRIPTION:

THE PRESENT STUDY ADDRESSES ITSELF TO THE PROBLEM OF DESIGNING AN AUTOMATED SYSTEM FOR INSTRUCTION IN PROGRAMMING, AND ALSO TO THE STUDY OF PROBLEM-SOLVING BEHAVIOR, AS EXHIBITED BY STUDENTS USING A CAI COURSE IN COMPUTER PROGRAMMING.

THE STUDY USES COMPUTER PROGRAMS WRITTEN BY 40 COLLEGE STUDENTS DURING THE WINTER AND SPRING QUARTERS OF 1972 AS PART OF A CAI COURSE IN AID (ALGEBRAIC INTERPRETIVE DIALOGUE), AN ALGEBRAIC LANGUAGE SIMILAR TO BASIC. THE COURSE IS SELF-CONTAINED AND CONSISTS OF 50 TUTORIAL LESSONS DESCRIBED IN DETAIL IN FRIEND (1973B).

THE PROGRAMS ANALYZED WERE WRITTEN AS SOLUTIONS TO 25 PROGRAMMING PROBLEMS FROM THE COURSE; 747 SOLUTIONS CONTAINING 7063 COMMANDS WERE ANALYZED. THE DISTRIBUTION OF THE DATA OVER PROBLEMS AND OVER STUDENTS IS DISCUSSED. PROBLEM DIFFICULTY AND DIVERSITY OF STUDENT SOLUTIONS ARE ALSO DISCUSSED IN DETAIL. (A)

270P, 7R.

122 COMPUTER-ASSISTED INSTRUCTION

FRIEND, J.E., FLETCHER, J.D., & ATKINSON, R.C. STUDENT PERFORMANCE IN COMPUTER-ASSISTED INSTRUCTION IN PROGRAMMING (TECHNICAL REPORT NO. 184). STANFORD, CALIFORNIA: STANFORD UNIVERSITY, INSTITUTE FOR MATHEMATICAL STUDIES IN THE SOCIAL SCIENCES, MAY 1972.

DESCRIPTION:

AN INSTRUCTIONAL SYSTEM FOR TEACHING ALGEBRAIC INTERPRETIVE DIALOGUE (AID) TO COLLEGE-AGE STUDENTS, TWO CONTROL PROGRAMS (ONE FOR PRESENTING INSTRUCTIONAL MATERIAL AND ONE FOR INTERPRETING STUDENTS' AID PRODUCTIONS), AND DATA COLLECTED BY THE TWO CONTROL PROGRAMS ARE DESCRIBED. THE FIRST 21 LESSONS OF THE COURSE AND CLASSIFICATION OF THE LESSON EXERCISES ARE ALSO DESCRIBED. DATA BASED ON STUDENT DAILY REPORTS ARE PRESENTED AND DISCUSSED. ITEM ANALYSES OF DATA GATHERED BY THE INSTRUCTIONAL PROGRAM, INCLUDING STEPWISE LINEAR REGRESSION MODELS OF ITEM DIFFICULTY AND ANALYSES OF SELECTED DATA COLLECTED BY THE INTERPRETING PROGRAM ARE PRESENTED AND DISCUSSED.

THE FOLLOWING WERE AMONG THE RESULTS OF THIS INVESTIGATION: ALTHOUGH FIRST RESPONSE ERRORS WERE OFTEN THOSE OF AID SYNTAX, THESE ERRORS WERE EASILY CORRECTED IN SUBSEQUENT RESPONSES, AND IN GENERAL, THE SYNTAX OF AID COMMANDS WAS EASILY MASTERED; ALTHOUGH STUDENTS EASILY LEARNED THE MECHANICS OF THE INSTRUCTIONAL SYSTEM, THEY RARELY USED FEATURES THAT ALLOWED STUDENT CONTROL OF INSTRUCTIONAL CONTENT AND SEQUENCE; ALGEBRAIC FORMULATION OF PROBLEMS APPEARED TO BE MORE DIFFICULT THAN TRANSFORMING ALGEBRAIC EXPRESSIONS INTO AID COMMANDS; STUDENTS HAD THE GREATEST DIFFICULTY UNDERSTANDING HIERARCHY OF ARITHMETIC OPERATIONS, USE OF FUNCTIONS, AND THE EXECUTION SEQUENCE OF AID COMMANDS. (A)

123 PROGRAMMING

FROST, D. PSYCHOLOGY AND PROGRAM DESIGN. DATAMATION, MAY 1975, 137-138.

DESCRIPTION:

THE PURPOSE OF THIS PAPER IS TO CONSIDER HOW PSYCHOLOGICAL THEORIES CAN BE USEFULLY APPLIED TO IMPROVING PROGRAMMING AND PROGRAM DESIGN. THIS PAPER CENTERS ON THE PSYCHOLOGICAL PHENOMENON OF 'CHUNKING', OR THE GROUPING OF RELATED PIECES OF INFORMATION INTO A SINGLE UNIT. IT HAS BEEN EXTENSIVELY DEMONSTRATED THAT THE AMOUNT OF INFORMATION THAT THE HUMAN MIND CAN RETAIN AT ANY ONE TIME IS APPROXIMATELY SEVEN "CHUNKS". CHUNKING EXTENDS THE AMOUNT OF INFORMATION THAT CAN BE RETAINED, THE NUMBER OF CHUNKS REMAINS CONSTANT. THE PROCESSES OF 'ABSTRACTION' AND 'DECOMPOSITION', WHICH TEND TO RELATE CHUNKS IN A HIERARCHIC FASHION ARE ALSO CONSIDERED. ON THE BASIS OF THESE RESULTS, ARGUMENTS ARE MADE IN FAVOR OF MODULARITY AND HIERARCHICAL DESIGN ('TOP-DOWN DECOMPOSITION'). (MEA)

2P, 0R.

124 SOFTWARE PHYSICS

FUNAMI, Y., & HALSTEAD, M.H. A SOFTWARE PHYSICS ANALYSIS OF AKIYAMA'S DEBUGGING DATA (TECHNICAL REPORT NO. CSD-TR-144). LAFAYETTE, INDIANA: PURDUE UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, 1975.

DESCRIPTION:

CORRELATION BETWEEN E (HALSTEAD'S "MENTAL EFFORT" MEASURE) AND NUMBER OF BUGS OBSERVED IN 9 MODULES REPRESENTING 84 MAN-MONTHS OF PROGRAMMING WAS .982. AKIYAMA'S OBSERVATIONS REPORTED WERE: COPRELATION BETWEEN BUGS AND SIZE OF MODULE WAS .83, CORRELATION BETWEEN BUGS AND DECISIONS AND CALLS WAS .92. PREDICTED TIME TO COMPLETE 9 MODULES WAS 100 MAN-MONTHS AS COMPARED WITH AN ACTUAL 84 MAN-MONTHS. (O)

4P, 7R.

125 DEBUGGING

GAINES, R.S. THE DEBUGGING OF COMPUTER PROGRAMS (DOCTORAL DISSERTATION, PRINCETON UNIVERSITY, 1969). (UNIVERSITY MICROFILMS NO. 7-14, 209; ALSO IDA WORKING PAPER NO. 226, INSTITUTE FOR DEFENSE ANALYSES)

DESCRIPTION:

THIS THESIS IS A GENERAL STUDY OF THE TOOLS AND TECHNIQUES FOR DEBUGGING COMPUTER PROGRAMS, AND THE DESIGN OF COMPILERS AND OPERATING SYSTEMS TO FACILITATE DEBUGGING. IT INCLUDES AN ANALYSIS OF THE PROGRAMMING PROCESS TO IDENTIFY CAREFULLY WHAT THE PROBLEMS ARE IN DEBUGGING AND HOW THEY ARISE. BASED ON THIS ANALYSIS, THE FUNDAMENTAL NOTIONS WHICH UNDERLIE POST DEBUGGING AIDS ARE IDENTIFIED. THE FACILITIES THAT ARE CURRENTLY AVAILABLE TO PROGRAMMERS USING BATCH OPERATING SYSTEMS ARE DISCUSSED, AND A NUMBER OF NEW ONES ARE PRESENTED.

THE PROBLEMS OF DEBUGGING PROGRAMS WRITTEN IN HIGHER-LEVEL LANGUAGES ARE CONSIDERED IN DETAIL, AND THE CONSTRUCTION OF COMPILERS AND PROGRAMMING LANGUAGES FOR DEBUGGING RECEIVES CAREFUL ATTENTION. IN THIS CONNECTION, THE TOPICS OF AUTOMATIC ERROR DETECTION, LANGUAGE FACILITIES WHICH PERMIT THE PROGRAMMER TO INVOKE DEBUGGING AIDS, AND THE COMPILATION OF CODE IN A MANNER APPROPRIATE FOR DEBUGGING THE PROGRAM ARE ALL DISCUSSED. A METHOD IS PROPOSED TO PERMIT THE HIGHER-LEVEL LANGUAGE PROGRAMMER TO DEBUG HIS PROGRAM AS IF THE LANGUAGE HE WROTE HIS PROGRAM IN WERE THE "MACHINE LANGUAGE" OF THE COMPUTER WHICH EXECUTES THE PROGRAM, AS A CONSEQUENCE OF WHICH THE PROGRAMMER CAN RECEIVE THE SAME KIND OF ASSISTANCE THAT IS AVAILABLE TO THE MACHINE LANGUAGE PROGRAMMER.

THE CONSTRUCTION OF INTERACTIVE DEBUGGING SYSTEMS AND OPERATING SYSTEM FEATURES NECESSARY TO SUPPORT ADVANCED DEBUGGING FACILITIES IS DISCUSSED. THREE INTERACTIVE DEBUGGING SYSTEMS ARE CONSIDERED IN DETAIL, INCLUDING ONE WHICH WAS DESIGNED TO WORK WITH CRT CONSOLES. A NUMBER OF NEW IDEAS IN THIS AREA ARE PRESENTED, AND THE CONSIDERABLE ADVANTAGES OF INTERACTIVE DEBUGGING ARE CLEARLY DEMONSTRATED. (A)

170P, 53R.

126 PROGRAMMING LANGUAGES

GANNON, J. LANGUAGE AND COMPILER DESIGN TO ENHANCE RELIABILITY. SIGPLAN NOTICES, 1973, 8(6), 47-49.

DESCRIPTION:

THIS BRIEF ARTICLE DESCRIBES THE OBJECTIVES AND INTENDED APPROACH OF A PROJECT TO ISOLATE THE FACTORS THAT MAKE PROGRAMS EASILY UNDERSTOOD AND THE CONSTRUCTS IN A LANGUAGE THAT ARE MOST PRONE TO CAUSE ERRORS. THE THESIS IS THAT A LANGUAGE SHOULD EMPLOY USEFUL AND CHECKABLE REDUNDANCY, MINIMIZE THE USE OF "UNDISCIPLINED CONSTRUCTS," AND EMPLOY CONSTRUCTS AMENABLE TO PROOFS. (GDC)

3P, 8R.

127 PROGRAMMING LANGUAGES

GANNON, J.D. LANGUAGE DESIGN TO ENHANCE PROGRAMMING RELIABILITY (TECHNICAL REPORT NO. CSRG-47). TORONTO, ONTARIO: UNIVERSITY OF TORONTO, COMPUTER SYSTEMS RESEARCH GROUP, 1975.

DESCRIPTION:

THE LANGUAGE IN WHICH PROGRAMS ARE WRITTEN CAN HAVE A SUBSTANTIAL EFFECT ON THEIR RELIABILITY. THIS THESIS DISCUSSES THE DESIGN OF PROGRAMMING LANGUAGES TO ENHANCE THE RELIABILITY OF PROGRAMS. IT PRESENTS SEVERAL DESIGN PRINCIPLES, AND THEN APPLIES THEM TO PARTICULAR LANGUAGE CONSTRUCTS. SINCE ONE CANNOT LOGICALLY PROVE THE VALIDITY OF SUCH DESIGN PRINCIPLES, EMPIRICAL EVIDENCE IS NEEDED TO SUPPORT OR DISCREDIT THEM. AN EXPERIMENT WAS PERFORMED TO MEASURE THE EFFECT OF NINE SPECIFIC LANGUAGE DESIGN DECISIONS IN ONE CONTEXT. ANALYSIS OF THE FREQUENCY AND PERSISTENCE OF ERRORS SHOWS THAT SEVERAL DECISIONS HAD A SIGNIFICANT IMPACT ON RELIABILITY. (A)

247P, 61R.

128 PROGRAMMING LANGUAGES
GANNON, J.D. AN EXPERIMENTAL EVALUATION OF DATA TYPE CONVENTIONS. COMMUNICATIONS OF THE ACM, 1977, 20, 584-595 (ALSO TECHNICAL REPORT, COLLEGE PARK, MARYLAND: UNIVERSITY OF MARYLAND, DEPARTMENT OF COMPUTER SCIENCE, 1976).
DESCRIPTION:
THE LANGUAGE IN WHICH PROGRAMS ARE WRITTEN CAN HAVE A SUBSTANTIAL EFFECT ON THE RELIABILITY OF THE RESULTING PROGRAMS. THIS PAPER DISCUSSES AN EXPERIMENT THAT COMPARES THE PROGRAMMING RELIABILITY OF SUBJECTS USING A STATICALLY-TYPED LANGUAGE AND A 'TYPELESS' LANGUAGE. ANALYSIS OF THE NUMBER OF ERRORS SHOWS THAT, AT LEAST IN ONE ENVIRONMENT, THE USE OF A STATICALLY-TYPED LANGUAGE CAN INCREASE PROGRAMMING RELIABILITY. DETAILED ANALYSIS OF THE ERRORS MADE BY THE SUBJECTS IN PROGRAMMING SOLUTIONS TO REASONABLY SMALL PROBLEMS SHOWS THAT THE SUBJECTS HAD DIFFICULTY MANIPULATING THE REPRESENTATION OF DATA. (A)
34P, 6R.

129 PROGRAMMING LANGUAGES
GANNON, J.D. AN EXPERIMENT FOR THE EVALUATION OF LANGUAGE FEATURES. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1976, 8, 61-73.
DESCRIPTION:
RECENTLY A NUMBER OF EXPERIMENTS HAVE BEEN PERFORMED WHOSE AIM WAS TO COMPARE PROGRAMMING LANGUAGE FEATURES TO DETERMINE WHICH PROGRAMMING LANGUAGE FEATURES PROGRAMMERS FOUND DIFFICULT TO USE. THIS PAPER EXAMINES THESE EXPERIMENTS IN LIGHT OF THE EVIDENCE THAT PROGRAMMING LANGUAGE DESIGNERS WOULD FIND MOST USEFUL. A NEW EXPERIMENT IS DESCRIBED AND APPLIED TO THE PROBLEM OF WHETHER THE ASSIGNMENT OPERATION SHOULD BE DEFINED AS AN OPERATOR OR A STATEMENT DESIGNATOR. EMPIRICAL EVIDENCE IN THE FORM OF ERRORS MADE BY STUDENTS PROGRAMMING SOLUTIONS TO TWO GOOD-SIZED PROBLEMS IS PRESENTED FAVORING THE USE OF ASSIGNMENT AS A STATEMENT. FINALLY, THE SHORTCOMINGS OF THE NEW EXPERIMENT ARE DISCUSSED. (A)
13P, 15R.

130 PROGRAMMING LANGUAGES
GANNON, J.D., & HORNING, J.J. LANGUAGE DESIGN FOR PROGRAMMING RELIABILITY. IEEE TRANSACTION ON SOFTWARE ENGINEERING, 1975, SE-1, 179-191 (ALSO: THE IMPACT OF LANGUAGE DESIGN ON THE PRODUCTION OF RELIABLE SOFTWARE. IN PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON RELIABLE SOFTWARE, 21-23 APRIL 1975, LOS ANGELES, CALIFORNIA. SIGPLAN NOTICES, 1975, 10(6), 10-22.)
DESCRIPTION:
THE LANGUAGE IN WHICH PROGRAMS ARE WRITTEN CAN HAVE A SUBSTANTIAL EFFECT ON THEIR RELIABILITY. THIS PAPER DISCUSSES THE DESIGN OF PROGRAMMING LANGUAGES TO ENHANCE RELIABILITY. IT PRESENTS SEVERAL GENERAL DESIGN PRINCIPLES, AND THEN APPLIES THEM TO PARTICULAR LANGUAGE CONSTRUCTS. SINCE WE CANNOT LOGICALLY PROVE THE VALIDITY OF SUCH DESIGN PRINCIPLES, EMPIRICAL EVIDENCE IS NEEDED TO SUPPORT OR DISCREDIT THEM. GANNON HAS PERFORMED A MAJOR EXPERIMENT TO MEASURE THE EFFECT OF NINE SPECIFIC LANGUAGE DESIGN DECISIONS IN ONE CONTEXT. ANALYSIS OF THE FREQUENCY AND PERSISTENCE OF ERRORS SHOWS THAT SEVERAL DECISIONS HAD A SIGNIFICANT IMPACT ON RELIABILITY. (A)
13P, 44R.

131 DOCUMENTATION
GENERAL ACCOUNTING OFFICE. IMPROVEMENT NEEDED IN DOCUMENTING COMPUTER SYSTEMS (REPORT NO. B-11536). OCTOBER, 1974.

132 PROGRAMMING
GERHART, S.L. KNOWLEDGE ABOUT PROGRAMS: A MODEL AND CASE STUDY. IN PROCEEDINGS, 1975 INTERNATIONAL CONFERENCE ON RELIABLE SOFTWARE. SIGPLAN NOTICES, JUNE 1975, 10(6), 88-95.
DESCRIPTION:
DIJKSTRA SUGGESTS IN HIS "NOTES ON STRUCTURED PROGRAMMING" THAT PROGRAM SCHEMA AND THEOREMS ABOUT THEIR CORRECTNESS MAY DESCRIBE THE WAY THAT PROGRAMMERS UNDERSTAND PROGRAMMING. THIS PAPER FOLLOWS UP HIS SUGGESTION BY DESCRIBING A GENERAL MODEL FOR DOMAINS OF PROGRAMMING KNOWLEDGE IN TERMS OF SCHEMA, TRANSFORMATIONS, AND DERIVATION RULES. THE MODEL IS ILLUSTRATED BY THE RESULTS OF A CASE STUDY OF THE KNOWLEDGE ABOUT 10 PROGRAMS WHICH USE ARRAYS. THE MODEL AND CASE STUDY ILLUSTRATE A METHODOLOGY FOR CONSTRUCTING AND PROVING CORRECT PROGRAMS BASED ON KNOWLEDGE WHICH IS INDEPENDENTLY EXPRESSED AND PARTIALLY PROVED IN AN ABSTRACT FORM AND WHICH CAN BE APPLIED IN A STEPWISE WAY. (A)
8P, 6R.

133 TESTING
GERHART, S.L. & YELOWITZ, L. OBSERVATIONS OF FALLIBILITY IN APPLICATIONS OF MODERN PROGRAMMING METHODOLOGIES. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1976, SE-2, 195-207.

134 PROGRAMMING
GIBBONS, G. PRODUCTION SYSTEM PROGRAMMING. COMMUNICATIONS OF THE ACM, 1976, 19, 105-106.
DESCRIPTION:
THIS LETTER IS INTENDED TO INTRODUCE THE PROGRAMMING LANGUAGE COMMUNITY TO THE CONCEPT OF PRODUCTION SYSTEM PROGRAMMING, THAT HAS BEEN USED PRIMARILY IN ARTIFICIAL INTELLIGENCE RESEARCH. FIRST, THE IDENTITY BETWEEN PRODUCTIONS AND THE "GUARDED COMMAND" PROPOSED BY E.W. DIJKSTRA (COMMUNICATIONS OF THE ACM, AUGUST 1975, 18(8), 453-457) WILL BE DEMONSTRATED AND THEN SOME OF THE USES AND CHARACTERISTICS OF PRODUCTION SYSTEM PROGRAMMING WILL BE MENTIONED. PRODUCTION SYSTEM PROGRAMMING MAY WELL BE THE KEY CONCEPT THAT WILL REVOLUTIONIZE PROGRAMMING METHODOLOGY; WHAT ALGEBRAIC LANGUAGES DID FOR FORMULA CALCULATION, PRODUCTION SYSTEM PROGRAMMING WILL DO FOR LOGICAL FLOW OF CONTROL. (MEA)
2P, 7R.

135 STRUCTURED PROGRAMMING
GILB, T. A SKEPTICAL VIEW OF STRUCTURED PROGRAMMING AND SOME ALTERNATIVES: PART I. COMPUTERS AND PEOPLE, MAY 1976, 25(5), 20-23.

136 STRUCTURED PROGRAMMING
GILB,T. A SKEPTICAL VIEW OF STRUCTURED PROGRAMMING AND SOME ALTERNATIVES: PART 2. COMPUTERS AND PEOPLE, JUNE 1976, 25(6), 21-22.
DESCRIPTION:
ALTHOUGH A GREAT DEAL HAS BEEN WRITTEN ABOUT THE RELATIVE ADVANTAGES AND MERITS OF STRUCTURED PROGRAMMING, LITTLE ATTENTION HAS BEEN GIVEN TO COMPARABLE, AND PERHAPS COMPLEMENTARY, TECHNIQUES. THIS PAPER DISCUSSES THE TECHNIQUES OF DUAL CODE AND PARALLEL PROGRAMMING, AUTOMATED TEST PATH ANALYSIS, PROCESS INSPECTIONS, AND DATA REDUNDANCY-BASED ERROR-DETECTION AND CORRECTION METHODS. (MEA)
3P, 24R.

137 SOFTWARE METRICS
GILB, T. SOFTWARE METRICS. CAMBRIDGE, MA: WINTHROP PUBLISHERS, INC., 1977.

138 PROGRAMMING
GILEADI, A.N., & LEDGARD, H.F. ON A PROPOSED MEASURE OF PROGRAM STRUCTURE. SIGPLAN NOTICES, MAY 1974, 9(5), 31-36.
DESCRIPTION:
AN IMPORTANT PROBLEM IN QUALITY SOFTWARE RESEARCH IS THE MEASUREMENT OF CERTAIN OBJECTIVE PROPERTIES OF PROGRAMS. WE ARE CONCERNED HERE WITH A MEASURE THAT WE HOPE WILL TELL US HOW "WELL-STRUCTURED" A PARTICULAR FLOWCHART IS WITH RESPECT TO THE PRECEPTS OF STRUCTURED PROGRAMMING.
WITH SUCH A MEASURE, WE COULD DETERMINE WHICH OF SEVERAL FLOWCHARTS USED TO DESCRIBE A PROGRAM IS BEST FROM A STRUCTURAL POINT OF VIEW, RATHER THAN FROM THE CLASSICAL VIEWPOINTS: THE LEAST COMPUTATION TIME OR THE LEAST AMOUNT OF STORAGE NEEDED. TO ARRIVE AT THE PROPOSED MEASURE, WE PROCEED THROUGH A SERIES OF TRANSFORMATIONS. THEY ARE DESCRIBED HEREIN MAINLY BY EXAMPLE. (A)
6P, 9R.

139 PROGRAMMING
GINSBERG, A.S., MARKOWITZ, H.M., & OLDFATHER, P.M. PROGRAMMING BY QUESTIONNAIRE (TECHNICAL REPORT NO. RM-4460-PR). SANTA MONICA, CALIFORNIA: THE RAND CORPORATION, 1965.
DESCRIPTION:
IT HAS BECOME CLEAR THAT GREAT AMOUNTS OF EFFORT AND TIME ARE REQUIRED TO PREPARE THE NECESSARY COMPUTER PROGRAMS IN THE USE OF DIGITAL COMPUTERS FOR SIMULATION STUDIES. A TECHNIQUE FOR REDUCING THIS EFFORT AND PROGRAM PREPARATION TIME HAS BEEN DEVELOPED. THIS TECHNIQUE, 'PROGRAMMING BY QUESTIONNAIRE' (OR THE PROGRAM GENERATION CONCEPT) ALLOWS A USER TO OBTAIN A SIMULATION PROGRAM BY FILLING OUT AN ENGLISH LANGUAGE QUESTIONNAIRE. THIS APPROACH HAS SUFFICIENT GENERALITY THAT IT PROMISES APPLICATION TO AREAS OF COMPUTER PROGRAMMING OTHER THAN SIMULATION.
THIS MEMORANDUM DESCRIBES THE QUESTIONNAIRE TECHNIQUE, COMPARES IT TO EXISTING TECHNIQUES, AND DISCUSSES POTENTIAL APPLICATIONS. THE WORKINGS OF THE TECHNIQUE ARE DESCRIBED IN TERMS OF THE JOB SHOP SIMULATION PROGRAM GENERATOR, AN EXAMPLE DEVELOPED TO TEST THE FEASIBILITY AND DESIRABILITY OF THE CONCEPT.
PROGRAMMING BY QUESTIONNAIRE SHOULD BE OF INTEREST TO ALL THOSE CONCERNED WITH DEVELOPING MAJOR COMPUTER PROGRAMS. (A)

140 TIME-SHARING

GOLD, M.M. A METHODOLOGY FOR EVALUATING TIME-SHARED COMPUTER SYSTEM USAGE. UNPUBLISHED DOCTORAL DISSERTATION, MASSACHUSETTS INSTITUTE OF TECHNOLOGY, CAMBRIDGE, MASSACHUSETTS, JUNE 1967.

DESCRIPTION:

THE DEVELOPMENT OF TIME-SHARED COMPUTER SYSTEMS HAS LED TO MAJOR TECHNICAL AND PHILOSOPHIC CHANGES IN THE COMPUTER FIELD THIS DECADE. A LARGE NUMBER OF DESIGNERS, MANUFACTURERS, AND USERS OF SUCH SYSTEMS HAVE EXPENDED GREAT AMOUNTS OF EFFORT IN THE DEVELOPMENT OF THE CAPABILITIES OF THE COMPUTER AND THE MEANS TO USE IT. HOWEVER, LITTLE OR NO EFFORT HAS YET BEEN EXPENDED TO EVALUATE THESE SYSTEMS IN TERMS OF THEIR USEFULNESS FOR PRESENT OR FUTURE CUSTOMERS.

THE RESEARCH REPORTED HERE HAS FOCUSED ON THE DEVELOPMENT OF A METHODOLOGY THROUGH WHICH TIME-SHARED COMPUTER SYSTEM USAGE CAN BE EVALUATED. IT IS BASED ON A STUDY OF THE CHARACTERISTICS AND DESIGN OF PRESENT AND PROPOSED COMPUTER SYSTEMS, AS WELL AS RELEVANT BEHAVIORAL THEORY AND RESEARCH. FOUR CATEGORIES OF VARIABLES ARE INCLUDED IN THE RESULTING METHODOLOGY, NAMELY THOSE WHICH ARE MEASURES OF: (1) THE COST OF USING THE SYSTEM, (2) THE PERFORMANCE PRODUCED THROUGH USE OF THE COMPUTER SYSTEM, (3) THE SPEED WITH WHICH RESULTS COULD BE PRODUCED, AND (4) THE AMOUNT OF LEARNING RESULTING FROM THE USE OF THE COMPUTER SYSTEM.

THE METHODOLOGY DEVELOPED WAS TESTED EXPERIMENTALLY THROUGH EVALUATING USAGE OF TWO COMPUTER SYSTEMS, EACH EXHIBITING CERTAIN CHARACTERISTICS OF BOTH TIME-SHARING AND BATCH-PROCESSING. THE PRIMARY PROBLEM UNDER STUDY WAS THE EFFECT OF RAPID FEEDBACK AND UNLIMITED COMPUTER ACCESS IN A PROBLEM-SOLVING SITUATION -- THE SECONDARY INVESTIGATION INVOLVED THE EFFECT OF QUALITATIVELY DIFFERENT FEEDBACK UPON COMPUTER PROGRAMMING. (A)

152P, 87R.

141 TIME-SHARING VERSUS BATCH PROCESSING

GOLD, M.M. TIME-SHARING AND BATCH-PROCESSING: AN EXPERIMENTAL COMPARISON OF THEIR VALUES IN A PROBLEM-SOLVING SITUATION. COMMUNICATIONS OF THE ACM, 1969, 12, 249-259.

DESCRIPTION:

AN EXPERIMENTAL COMPARISON OF PROBLEM-SOLVING USING TIME-SHARING AND BATCH-PROCESSING COMPUTER SYSTEMS CONDUCTED AT MIT IS DESCRIBED IN THIS PAPER. THIS STUDY IS THE FIRST KNOWN ATTEMPT TO EVALUATE TWO SUCH SYSTEMS FOR WHAT MAY WELL BE THE PREDOMINANT USER POPULATION WITHIN THE NEXT DECADE -- THE PROFESSIONALS WHO, AS NONPROGRAMMERS, ARE USING THE COMPUTER AS AN AID IN DECISION-MAKING AND PROBLEM-SOLVING RATHER THAN AS A PROGRAMMING END IN ITSELF.

STATISTICALLY AND LOGICALLY SIGNIFICANT RESULTS INDICATE EQUAL COST FOR USAGE OF THE TWO COMPUTER SYSTEMS; HOWEVER, A MUCH HIGHER LEVEL OF PERFORMANCE IS ATTAINED BY TIME-SHARING USERS. THERE ARE INDICATIONS THAT SIGNIFICANTLY LOWER COSTS WOULD HAVE RESULTED IF THE TIME-SHARING USERS HAD STOPPED WORK WHEN THEY REACHED A PERFORMANCE LEVEL EQUAL TO THAT OF THE BATCH USERS. THE USERS' SPEED OF PROBLEM-SOLVING AND THEIR ATTITUDES MADE TIME-SHARING THE MORE FAVORABLE SYSTEM. (A)

11P, 28R.

142 TIME-SHARING

GOLD, M.M., & STEDRY, A.C., AN EVALUATION OF COMMERCIAL TIME SHARING SYSTEMS. MAY, 1966. (NTIS NO. AD 634325)

143 SOFTWARE DEVELOPMENT

GOLDBERG, J. (ED.) PROCEEDINGS OF A SYMPOSIUM ON THE HIGH COST OF SOFTWARE. MENLO PARK, CALIFORNIA: STANFORD RESEARCH INSTITUTE, SEPTEMBER 1973.

DESCRIPTION:

THE MONTEREY SYMPOSIUM ON THE HIGH COST OF SOFTWARE WAS HELD IN SEPTEMBER 1973, UNDER THE JOINT SPONSORSHIP OF THE AIR FORCE OFFICE OF SCIENTIFIC RESEARCH, THE ARMY RESEARCH OFFICE, AND THE OFFICE OF NAVAL RESEARCH. THE OBJECTIVE OF THE SYMPOSIUM WAS TO CONSIDER WHAT RESEARCH IS NEEDED TO ACHIEVE A MAJOR REDUCTION IN SOFTWARE COSTS. ATTENDANCE WAS BY INVITATION. THE 97 ATTENDEES WERE ORGANIZED IN FIVE WORKSHOPS.

THE ATTENDEES WERE IN STRONG AGREEMENT THAT DIRECT AND INDIRECT SOFTWARE COSTS ARE UNNECESSARILY HIGH AND ARE GROWING RAPIDLY, THAT THEY CONSTITUTE A SERIOUS LIMITATION ON THE EFFECTIVENESS OF INFORMATION-PROCESSING SYSTEMS, AND THAT THE HIGH COST IS A CONSEQUENCE OF THE POOR STATE OF THE ART OF SOFTWARE DESIGN, PRODUCTION, AND MAINTENANCE. THERE WAS A STRONG FEELING OF URGENCY THAT AN ENERGETIC PROGRAM OF RESEARCH BE UNDERTAKEN TO ADVANCE THE SOFTWARE ART.

THE WORKSHOP DISCUSSIONS RESULTED IN TWO SETS OF RECOMMENDATIONS FOR A SERVICE-SUPPORTED RESEARCH PROGRAM.

THE GENERAL RECOMMENDATIONS FOR THE OBJECTIVES OF THE PROGRAM WERE TO:

*OVERCOME PROBLEMS AT THE CRUCIAL INTERFACES OF COMPUTER SCIENCE, APPLICATIONS, AND COMPUTER HARDWARE AND SYSTEMS.
*FOSTER THE DEVELOPMENT OF A STRENGTHENED SERVICE TECHNOLOGY BASE IN SOFTWARE AND COMPUTER SYSTEMS.
*COORDINATE SERVICE-SUPPORTED RESEARCH WITH OTHER GOVERNMENT-SUPPORTED RESEARCH AND DEVELOPMENT.
*INCREASE THE SCALE AND QUALITY OF COMPUTER RESEARCH TO MEET PRESENT AND FUTURE DEMANDS.

TEN SPECIFIC TECHNICAL AREAS OF RESEARCH WERE RECOMMENDED IN THREE MAJOR CATEGORIES:

*TECHNIQUES FOR IMPROVING CURRENT PROGRAMMING PRACTICE.
*TECHNIQUES FOR ADVANCING PROGRAMMING METHODOLOGY.
*TECHNIQUES FOR PROGRAM GENERATION AND COMPUTER SYSTEM DESIGN.

THE PROCEEDINGS OF THE SYMPOSIUM CONTAIN REPORTS BY THE WORKSHOP CHAIRMAN IN FIVE AREAS:

*UNDERSTANDING THE SOFTWARE PROBLEM (CHAIRED BY JOHN B. SLAUGHTER)
*CONTRIBUTIONS FROM THE SEMANTICS OF LANGUAGES AND SYSTEMS (JACK B. DENNIS)
*ADVANCES IN SOFTWARE METHODOLOGY (WILLIAM A. WULF)
*SOFTWARE-RELATED ADVANCES IN COMPUTER HARDWARE (UGO O. GAGLIARDI)
*APPROACHES TO THE PROBLEMS OF LARGE SYSTEMS (JAMES H. BURROWS)

THESE PROCEEDINGS ALSO INCLUDE A SUMMARY OF A KEYNOTE SPEECH ON THE HIGH COST OF SOFTWARE AND A REPORT OF A PANEL ON SOFTWARE TECHNOLOGY TRANSFER.

DIGESTS OF THE REPORTS AND THE SPEECH ARE INCLUDED, TOGETHER WITH SUMMARIES OF THE RECOMMENDATIONS OF THE INDIVIDUAL WORKSHOPS. (A)
138P, 39R.

144 SOFTWARE TESTING AND VALIDATION

GOOD, D.I., LONDON, R.L., & BLEDSOE, W.W. INTERACTIVE PROGRAM VERIFICATION SYSTEM. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 59-67.

145 TESTING

GOODENOUGH, J.B., & GERHART, S.L. TOWARD A THEORY OF TEST DATA SELECTION. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 156-173 (ALSO IN PROCEEDINGS OF 1975 INTERNATIONAL CONFERENCE ON RELIABLE SOFTWARE, 21-23 APRIL 1975, LOS ANGELES, CALIFORNIA. SIGPLAN NOTICES, JUNE 1975, 10(6), 493-510).

DESCRIPTION:

THIS PAPER EXAMINES THE THEORETICAL AND PRACTICAL ROLE OF TESTING IN SOFTWARE DEVELOPMENT. WE PROVE A FUNDAMENTAL THEOREM SHOWING THAT PROPERLY STRUCTURED TESTS ARE CAPABLE OF DEMONSTRATING THE ABSENCE OF ERRORS IN A PROGRAM. THE THEOREM'S PROOF HINGES ON OUR DEFINITION OF TEST RELIABILITY AND VALIDITY, BUT ITS PRACTICAL UTILITY HINGES ON BEING ABLE TO SHOW WHEN A TEST IS ACTUALLY RELIABLE. WE EXPLAIN WHAT MAKES TESTS UNRELIABLE (FOR EXAMPLE, WE SHOW BY EXAMPLE WHY TESTING ALL PROGRAM STATEMENTS, PREDICATES, OR PATHS IS NOT USUALLY SUFFICIENT TO INSURE TEST RELIABILITY), AND WE OUTLINE A POSSIBLE APPROACH TO DEVELOPING RELIABLE TESTS. WE ALSO SHOW HOW THE ANALYSIS REQUIRED TO DEFINE RELIABLE TESTS CAN HELP IN CHECKING A PROGRAM'S DESIGN AND SPECIFICATIONS AS WELL AS IN PREVENTING AND DETECTING IMPLEMENTATION ERRORS. (A)

18P, 16R.

146 STRUCTURED PROGRAMMING

GORDON, E.K. EXPERIENCE AND ACCOMPLISHMENTS WITH STRUCTURED PROGRAMMING. COMPUTER MAGAZINE, JUNE 1975, 8(6), 50-53.

DESCRIPTION:

A LARGE SOFTWARE PROJECT WAS IMPLEMENTED USING STRUCTURED PROGRAMMING TECHNIQUES. THREE SHORTCOMINGS WERE NOTED: EACH COMPONENT WENT THROUGH A PERIOD OF LOW VISIBILITY WHILE THE DETAILED DESIGN AND CODING WERE BEING DONE, PROBLEMS THAT WOULD RENDER THE SYSTEM ORGANIZATION INVALID WERE OFTEN NOT DETECTED UNTIL WELL INTO THE IMPLEMENTATION PHASE, AND THERE WAS NO EASY WAY TO RECOGNIZE WHAT GENERAL UTILITY ROUTINES WERE NEEDED. TO AVOID THESE DIFFICULTIES, A PROGRAM DESIGN LANGUAGE (PDL) WAS USED AS THE PRINCIPAL TOOL IN THE DESIGN PHASE. RESULTS INDICATE THAT THE QUALITY OF THE CODE INCREASED WHILE COST AND TIME FACTORS DECREASED. (MEA)

4P, 0R.

147 PROGRAMMING

GORDON, R.D., & HALSTEAD, M.H. AN EXPERIMENT COMPARING FORTRAN PROGRAMMING TIMES WITH THE SOFTWARE PHYSICS HYPOTHESIS. AFIPS CONFERENCE PROCEEDINGS, 1976, 45, 935-937 (ALSO: (TECHNICAL REPORT NO. CSD-TR 167). LAFAYETTE, INDIANA: PURDUE UNIVERSITY, OCTOBER 1975).

DESCRIPTION:

THIS STUDY TESTED A "SOFTWARE PHYSICS" FORMULA WHICH ATTEMPTS TO RELATE THE TIME REQUIRED TO IMPLEMENT A SOFTWARE ALGORITHM TO INFORMATION ABOUT OPERATOR AND OPERAND USAGE IN THE PROGRAM. THE AUTHORS CONTEND THAT THE FORMULA CORRESPONDS TO THE MENTAL EFFORT ASSOCIATED WITH THE CONSTRUCTION OF THE ALGORITHM. AN EXPLANATION OF THE FORMULA IS GIVEN. ONE OF THE AUTHORS IMPLEMENTED 11 PROGRAMS BASED ON PROGRAM SPECIFICATIONS PUBLISHED IN TEXTBOOKS, RECORDING THE TIME REQUIRED FOR EACH IMPLEMENTATION. THE FORMULA WAS THEN APPLIED TO EACH RESULTING PROGRAM. AVERAGE OBSERVED TIME WAS 35 MINUTES, WHILE AVERAGE PREDICTED TIME WAS 34.1 MINUTES (RANGE 5-92 MINUTES). CORRELATION OF PREDICTED WITH OBSERVED TIME WAS .934, WHEREAS CORRELATION OF PROGRAM LENGTH WITH OBSERVED TIME WAS .887. (HRR)

3P, 28R.

148 SOFTWARE DEVELOPMENT

GORDON, R.L., & LAMB, J.C. A CLOSE LOOK AT BROOK'S LAW. DATAMATION, JUNE 1977, 23(6), 81-86.

149 DEBUGGING

GOULD, J.D. SOME PSYCHOLOGICAL EVIDENCE ON HOW PEOPLE DEBUG COMPUTER PROGRAMS. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1975, 7, 151-182 (ALSO: (RESEARCH REPORT NO. RC-4542). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, SEPTEMBER 1973).

DESCRIPTION:

TEN EXPERIENCED PROGRAMMERS WERE EACH GIVEN THE SAME 12 FORTRAN LISTINGS TO DEBUG. EACH LISTING CONTAINED A NON-SYNTACTIC ERROR IN ONE LINE. MEDIAN DEBUG TIMES (7 MINUTES), NUMBER OF BUGS NOT FOUND (11% OF THE LISTINGS), AND NUMBER OF INCORRECT ASSERTIONS ABOUT THE LOCATION OF THE BUG (LESS THAN ONE PER LISTING) ALL REPLICATED EARLIER RESULTS (GOULD & DRONGOWSKI, 1974). ALTHOUGH SUBJECTS WERE GIVEN THE OPPORTUNITY TO USE THE INTERACTIVE DEBUGGING FACILITIES OF AN ON-LINE COMPUTER THEY RARELY DID SO. BUGS IN ASSIGNMENT STATEMENTS WERE ABOUT THREE TIMES AS HARD TO DETECT AS ARRAY OR ITERATION BUGS. DEBUGGING WAS ABOUT THREE TIMES AS EFFICIENT ON PROGRAMS SUBJECTS HAD DEBUGGED PREVIOUSLY (ALTHOUGH WITH A DIFFERENT BUG). A NUMBER OF BASIC CONCEPTS RELATING TO DEBUGGING ARE IDENTIFIED AND A GROSS THEORY OF DEBUGGING IS DESCRIBED. (A)

32P, 25R.

150 QUERY LANGUAGES

GOULD, J.D., & ASCHER, R.N. USE OF AN IQF-LIKE QUERY LANGUAGE BY NON-PROGRAMMERS (RESEARCH REPORT NO. RC-5279). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, FEBRUARY 1975 (ALSO PRESENTED AT MEETING OF THE AMERICAN PSYCHOLOGICAL ASSOCIATION, NEW ORLEANS, LOUISIANA, SEPTEMBER 1974). (NTIS NO. AD AD43028)

DESCRIPTION:

THIS EXPLORATORY EXPERIMENT ATTEMPTS TO EXAMINE SEPARATELY THE FORMULATION, PLANNING, AND CODING OF QUERIES. COLLEGE STUDENTS AND FILE CLERKS REQUIRED ABOUT TEN HOURS TO LEARN A QUERY LANGUAGE WHICH WAS SOMEWHAT SIMILAR TO IBM'S IQF QUERY LANGUAGE, BUT CONTAINED MORE FUNCTIONS. THEY WERE THEN GIVEN 15 TEST PROBLEMS THAT VARIED IN COMPLEXITY AND HOW WELL THEY WERE EXPRESSED. SUBJECTS WERE REQUIRED TO FORMULATE, THEN TO PLAN (WRITING EACH IN THEIR OWN WORDS), AND FINALLY TO CODE EACH PROBLEM. RESULTS PROVIDE SOME SUGGESTIONS ABOUT WHICH PROBLEM VARIABLES AFFECTED WHICH "STAGES" IN WRITING QUERIES. FOR EXAMPLE, WHETHER OR NOT A PROBLEM WAS WELL EXPRESSED SEEMED TO AFFECT PROBLEM FORMULATION TIME, BUT HAD NO EFFECT UPON PROBLEM PLANNING OR PROBLEM CODING TIMES. SPECIFIC LANGUAGE CONSTRUCTIONS (ADDITIONS TO IQF), SUCH AS CONTEXTUAL REFERENCING AND A NEW METHOD TO HANDLE LIMITED DISJUNCTIVE PROBLEMS, WERE SHOWN TO BE USEFUL. THE TYPES OF CODING ERRORS THAT SUBJECTS MADE WERE IDENTIFIED AND DISCUSSED. (A)

30P, 15R.

151 PROGRAMMING, BIBLIOGRAPHY

GOULD, J.D., DOHERTY, W.J., & BOIES, S.J., BIBLIOGRAPHY OF BEHAVIORAL ASPECTS OF ON-LINE COMPUTER PROGRAMMING (TECHNICAL REPORT RC 3513). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, AUGUST 1971.

DESCRIPTION:

THIS WORKING, NON-EXHAUSTIVE BIBLIOGRAPHY IS MEANT TO BE USEFUL FOR THOSE CONCERNED WITH HOW PEOPLE PROGRAM COMPUTERS. STUDIES THAT CONTAIN DATA ON HOW PEOPLE PROGRAM ARE EMPHASIZED. (A, ABBR.)

11P, 58R.

152 DEBUGGING

GOULD, J.D., & DRONGOWSKI, P. AN EXPLORATORY STUDY OF COMPUTER PROGRAM DEBUGGING. HUMAN FACTORS, 1974, 16, 258-277.

DESCRIPTION:

THIS EXPERIMENT REPRESENTS A NEW APPROACH TO THE STUDY OF THE PSYCHOLOGY OF PROGRAMMING, AND DEMONSTRATES THE FEASIBILITY OF STUDYING AN ISOLATED PART OF THE PROGRAMMING PROCESS IN THE LABORATORY. THIRTY EXPERIENCED FORTRAN PROGRAMMERS DEBUGGED 12 ONE-PAGE FORTRAN LISTINGS, EACH OF WHICH WAS SYNTACTICALLY CORRECT BUT CONTAINED ONE NON-SYNTACTIC ERROR (BUG). THREE CLASSES OF BUGS (ARRAY BUGS, ITERATION BUGS, AND BUGS IN ASSIGNMENT STATEMENTS) IN EACH OF FOUR DIFFERENT PROGRAMS WERE DEBUGGED. THE PROGRAMMERS WERE DIVIDED INTO FIVE GROUPS, BASED UPON THE INFORMATION, OR DEBUGGING "AIDS", GIVEN THEM. KEY RESULTS WERE THAT DEBUG TIMES WERE SHORT (MEDIAN = 6 MIN.). THE AIDS GROUPS DID NOT DEBUG FASTER THAN THE CONTROL GROUP -- PROGRAMMERS ADOPTED THEIR DEBUGGING STRATEGIES BASED UPON THE INFORMATION AVAILABLE TO THEM. THE RESULTS SUGGEST THAT PROGRAMMERS OFTEN IDENTIFY THE INTENDED STATE OF A PROGRAM BEFORE THEY FIND THE BUG. ASSIGNMENT BUGS WERE MORE DIFFICULT TO FIND THAN ARRAY AND ITERATION BUGS, PROBABLY BECAUSE THE LATTER COULD BE DETECTED FROM A HIGH-LEVEL UNDERSTANDING OF THE PROGRAMMING LANGUAGE ITSELF. DEBUGGING WAS AT LEAST TWICE AS EFFICIENT THE SECOND TIME PROGRAMMERS DEBUGGED A PROGRAM (THOUGH WITH A DIFFERENT BUG IN IT). A SIMPLE HIERARCHICAL DESCRIPTION OF DEBUGGING WAS SUGGESTED, AND SOME POSSIBLE "PRINCIPLES" OF DEBUGGING WERE IDENTIFIED. (A)
20P, 30R.

153 PROCEDURE SPECIFICATION

GOULD, J.D., LEWIS, C., & BECKER, C.A. WRITING AND FOLLOWING PROCEDURAL, DESCRIPTIVE, AND RESTRICTED SYNTAX LANGUAGE INSTRUCTIONS (RESEARCH REPORT NO. RC-5943). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, APRIL 1976. (NTIS NO. AD A041289)

DESCRIPTION:

TWO EXPLORATORY EXPERIMENTS COMPARED THE WAY PEOPLE (WITH NO EXPERIENCE IN THE USE OF COMPUTING SYSTEMS) WRITE AND CARRY OUT NATURAL LANGUAGE PROCEDURES, NATURAL LANGUAGE DESCRIPTIONS, AND INSTRUCTIONS EXPRESSED IN AN ARTIFICIAL RESTRICTED SYNTAX LANGUAGE. THE RESULTS SUGGEST THAT THERE IS NO SINGLE "NATURAL" WAY THAT PEOPLE WRITE SIMPLE PLANS AND INSTRUCTIONS. SPEED AND ACCURACY OF WRITING WERE ABOUT THE SAME FOR ALL THREE APPROACHES, ALTHOUGH THE LINGUISTIC CHARACTERISTICS DIFFERED GREATLY FROM APPROACH TO APPROACH. WHILE SUBJECTS WERE TOLERANT OF AMBIGUITY BOTH IN WRITING AND IN CARRYING OUT INSTRUCTIONS, THEY OFTEN VOLUNTARILY EMPLOYED RESTRICTED-SYNTAX NOTATION IN THEIR WRITING AFTER BEING EXPOSED TO THE NOTATION. SUBJECTS' ACCURACY IN FOLLOWING DETAILED INSTRUCTIONS WAS NO GREATER THAN THAT IN WRITING THOSE INSTRUCTIONS. (A)
24P, 0R.

154 LINE PRINTER OUTPUT FORMATTING

GRACE, G.L. APPLICATION OF EMPIRICAL METHODS TO COMPUTER-BASED SYSTEM DESIGN. JOURNAL OF APPLIED PSYCHOLOGY, 1966, 50, 442-450.

DESCRIPTION:

THIS STUDY PROVIDES INFORMATION ABOUT THE CLARITY AND USEFULNESS OF PRINTOUT FORMATS DESIGNED FOR USE BY MILITARY NONPROGRAMMER PERSONNEL. THREE PRINTOUT FORMATS CONTAINING THE SAME INFORMATION WERE DESIGNED. VERBAL PRINTOUT FORMAT PRESENTED INFORMATION IN WORDS; DATA BLOCK PRINTOUT FORMAT, IN SETS OF DATA; EIDOFORM PRINTOUT FORMAT IN MAPLIKE FORM. TWENTY-THREE MEN STATIONED AT PHOENIX AIR DEFENSE SECTOR SERVED AS SUBJECTS. IMMEDIATELY FOLLOWING THE EXPERIMENTAL SESSIONS, ATTITUDE INFORMATION WAS COLLECTED IN INDIVIDUAL INTERVIEWS. PRINTOUT FORMATS AND SETS OF INTERPRETATION QUESTIONS WERE COMBINED FOR ANALYSIS USING A LATIN-SQUARE DESIGN. ANALYSIS OF VARIANCE SHOWED EXPERIMENTAL TREATMENT CONDITIONS, PRINTOUT FORMATS, AND PRACTICE EFFECTS TO BE STATISTICALLY SIGNIFICANT. DIFFERENCES DUE TO SEQUENCE AND TEST FORMS WERE NOT SIGNIFICANT. ATTITUDE RESULTS SUPPORTED INFORMATION MEASURE FINDINGS. (A)

BOTH TEST SCORES AND SUBJECT PREFERENCE RATINGS WERE SLIGHTLY HIGHER FOR DATA BLOCK FORMAT THAN FOR VERBAL FORMAT AND BOTH WERE SUPERIOR, ON BOTH TEST SCORES AND PREFERENCE RATINGS, TO EIDOFORM FORMAT. (MEA)

9P, 11R.

155 GENERAL

GRACE, G.L. PREFACE TO HUMAN FACTORS SPECIAL ISSUE: HUMAN FACTORS IN INFORMATION PROCESSING SYSTEMS. HUMAN FACTORS, 1970, 12, 161-164.

156 TIME-SHARING VERSUS BATCH PROCESSING

GRANT, E.E. AN EMPIRICAL COMPARISON OF ON-LINE AND OFF-LINE DEBUGGING (REPORT NO. SP-2441). SANTA MONICA, CALIFORNIA: SYSTEM DEVELOPMENT CORP., MAY 1966. (NTIS NO. AD 633907)

DESCRIPTION:

THIS PAPER REPORTS THE RESULTS OF AN EXPERIMENT CONDUCTED AT SYSTEM DEVELOPMENT CORPORATION IN 1965 AND EARLY 1966. THE EXPERIMENT COMPARED THE PROGRAM DEBUGGING (CHECKOUT) PERFORMANCE OF PROGRAMMERS USING SDC'S TIME-SHARING SYSTEM (TSS) WITH THE DEBUGGING PERFORMANCE OF PROGRAMMERS USING A SIMULATED CLOSED SHOP. TWELVE PROGRAMMERS PARTICIPATED IN THE STUDY. EACH PROGRAMMER WAS GIVEN TWO PROBLEM STATEMENTS AND WAS ASKED TO WRITE A PROGRAM TO SOLVE EACH. ONE PROBLEM REQUIRED A PROGRAM TO INTERPRET AND SOLVE ALGEBRAIC EQUATIONS. THE OTHER PROBLEM REQUIRED A PROGRAM TO FIND THE SINGLE PATH THROUGH A 20X20 CELL MAZE REPRESENTED IN THE COMPUTER BY A 400 ENTRY TABLE. SIX SOLUTIONS (PROGRAMS) TO EACH PROBLEM WERE DEBUGGED ON-LINE USING TSS AND SIX WERE DEBUGGED OFF-LINE USING A SIMULATED CLOSED-SHOP SYSTEM WITH A DESK-TO-DESK TURNAROUND TIME OF TWO HOURS. PERFORMANCE WAS MEASURED IN TERMS OF MAN HOURS TO DEBUG AND CENTRAL PROCESSOR TIME USED IN DEBUGGING. PROGRAMMERS WHO DEBUGGED THEIR ALGEBRAIC INTERPRETATION PROGRAMS ON-LINE USED SLIGHTLY FEWER MAN HOURS AND ABOUT THREE TIMES AS MUCH CENTRAL PROCESSOR TIME AS DID PROGRAMMERS WHO DEBUGGED THESE PROGRAMS OFF-LINE. PROGRAMMERS WHO DEBUGGED THEIR MAZE PROGRAMS ON-LINE USED ABOUT ONE-THIRD AS MANY MAN HOURS AND SLIGHTLY MORE CENTRAL PROCESOR TIME THAN THOSE WHO DEBUGGED THEIR MAZE PROGRAMS OFF-LINE. RESULTS ARE DISCUSSED AND THE FOLLOWING POINTS ARE COVERED IN AN EFFORT TO RECONCILE THE DISPARATE RESULTS FROM THE TWO KINDS OF PROGRAMS: (1) THE ADEQUACY AND REALISM OF OFF-LINE SIMULATION, (2) INEQUALITIES BETWEEN THE GROUPS ASSIGNED TO EXPERIMENTAL CONDITIONS, (3) QUALITATIVE DIFFERENCES BETWEEN THE TWO KINDS OF EXPERIMENTAL PROBLEMS, (4) POSSIBLE ADOPTION OF INEFFICIENT WORK HABITS ON THE PART OF PROGRAMMERS WORKING ON-LINE. (A)

16P, 8R.

157 TIME-SHARING VS. BATCH PROCESSING

GRANT, E.E., & SACKMAN, H. AN EXPLORATORY INVESTIGATION OF PROGRAMMER PERFORMANCE UNDER ON-LINE AND OFF-LINE CONDITIONS. IEEE TRANSACTIONS ON HUMAN FACTORS IN ELECTRONICS, 1967, HFE-8, 33-48.

DESCRIPTION:

THIS IS THE FIRST KNOWN STUDY COMPARING THE PERFORMANCE OF PROGRAMMERS UNDER CONTROLLED CONDITIONS FOR A STANDARD TASK. AN EXPERIMENT WAS CONDUCTED TO COMPARE THE PERFORMANCE OF PROGRAMMERS WORKING UNDER CONDITIONS OF ON-LINE AND OFF-LINE ACCESS TO THE COMPUTER. TWO GROUPS OF SIX PROGRAMMERS EACH, COMPRISING A SAMPLE OF 12 SUBJECTS, CODED AND DEBUGGED TWO TYPES OF PROBLEMS UNDER ON-LINE AND OFF-LINE CONDITIONS IN ACCORDANCE WITH A LATIN-SQUARE EXPERIMENTAL DESIGN. THE ON-LINE CONDITION WAS THE NORMAL MODE OF OPERATION FOR THE SYSTEM DEVELOPMENT CORPORATION TIME-SHARING SYSTEM; THE OFF-LINE CONDITION WAS SIMULATED USING A TWO-HOUR TURNAROUND TIME.

STATISTICALLY SIGNIFICANT RESULTS INDICATED FASTER DEBUGGING UNDER ON-LINE CONDITIONS. PERHAPS THE MOST IMPORTANT PRACTICAL FINDING OF THIS STUDY, OVERSHADOWING ON-LINE/OFF-LINE DIFFERENCES, CONCERNED THE LARGE AND STRIKING INDIVIDUAL DIFFERENCES IN PROGRAMMER PERFORMANCE. ATTEMPTS ARE MADE TO RELATE OBSERVED INDIVIDUAL DIFFERENCES TO OBJECTIVE MEASURES OF PROGRAMMER EXPERIENCE AND PROFICIENCY THROUGH FACTORIAL TECHNIQUES. IN LINE WITH THE EXPLORATORY OBJECTIVES OF THIS STUDY, METHODOLOGICAL PROBLEMS ENCOUNTERED IN DESIGNING AND CONDUCTING THIS TYPE OF EXPERIMENT ARE DESCRIBED, LIMITATIONS OF THE FINDINGS ARE POINTED OUT, HYPOTHESES ARE PRESENTED TO ACCOUNT FOR THE RESULTS, AND SUGGESTIONS ARE MADE FOR FURTHER RESEARCH. (A)

16P, 13R.

158 PROGRAMMING LANGUAGES

GREEN, T.R.G. CONDITIONAL PROGRAM STATEMENTS AND THEIR COMPREHENSIBILITY TO PROFESSIONAL PROGRAMMERS. JOURNAL OF OCCUPATIONAL PSYCHOLOGY, 1977, 50, 93-109.

DESCRIPTION:

PROGRAMMING LANGUAGES EMBODY TWO PRINCIPAL FORMS OF CONDITIONAL STATEMENT, THE NESTING IF...THEN...ELSE... FORM AND THE GOTO FORM. SIME, GREEN & GUEST (1974) COMPARED THESE TWO AND A THIRD VARIETY NOT IN COMMON USE, NESTING WITH REDUNDANCY, AND THEY FOUND THAT NON-PROGRAMMERS LEARNING TO WRITE PROGRAMS BASED ON CONDITIONALS OBTAINED BEST RESULTS WITH THIS LAST VARIETY. THEIR EXPLANATION EMPHASIZED THE NEED TO COMPREHEND A PROGRAM, AND DISTINGUISHED BETWEEN TWO COMPREHENSION PROCESSES APPLICABLE TO CONDITIONALS: TRACING THROUGH A PROGRAM LIKE A COMPUTER, AND FINDING HOW A PARTICULAR POINT IN A PROGRAM MIGHT BE REACHED. THEY HYPOTHESIZED THAT COMPREHENSIBILITY DIFFERENCES OCCURRED MAINLY IN THE SECOND PROCESS. THE PRESENT EXPERIMENTS COMPARED RESPONSE TIMES OF PROFESSIONAL PROGRAMMERS IN COMPREHENSION TASKS REQUIRING EITHER THE FIRST PROCESS (EXPT. I) OR THE SECOND (EXPT. II). MUCH LARGER DIFFERENCES BETWEEN THE THREE STYLES OF CONDITIONAL STATEMENTS WERE FOUND IN THE SECOND TASK, FAVOURING NESTING WITH REDUNDANCY AND THUS SUPPORTING THE HYPOTHESIS. IMPLICATIONS FOR RESEARCH IN THIS AREA AND FOR PROGRAMMING LANGUAGE DESIGN ARE DISCUSSED. (A)

17P, 19R.

159 PROGRAMMING LANGUAGES
GREEN, T.R.G., SIME, M.E., & FITTER, M. BEHAVIOURAL EXPERIMENTS ON PROGRAMMING LANGUAGES: SOME METHODOLOGICAL CONSIDERATIONS (MRC MEMO NO. 66). SHEFFIELD, ENGLAND: SHEFFIELD UNIVERSITY, DEPARTMENT OF PSYCHOLOGY, 1975.
DESCRIPTION:
THIS PAPER OFFERS AN INTRODUCTION TO EMPIRICAL RESEARCH ON PROGRAMMING BEHAVIOUR AND THE DESIGN OF PROGRAMMING LANGUAGES. EMPIRICAL STUDIES ARE THE FASTEST ROUTE TO BETTER PROGRAMMING LANGUAGES -- ESPECIALLY TO LANGUAGES FOR CASUAL COMPUTER USERS. HOWEVER, EXPERIMENTS ON PROGRAMMING BEHAVIOUR PRESENT UNUSUAL PROBLEMS, AT ALL LEVELS FROM A THEORETICAL ANALYSIS OF WHAT IT MEANS TO UNDERSTAND A PROGRAM TO THE INVENTION OF A SCENARIO TO ASSIST THE EXPERIMENTAL SUBJECT. THE PROBLEMS ARE DISCUSSED FROM THE VIEWPOINT OF APPLIED PSYCHOLOGISTS, INTRODUCING RELEVANT EVIDENCE FROM COGNITIVE PSYCHOLOGY WHILE REVIEWING EXISTING LITERATURE ON EMPIRICAL PROGRAMMING LANGUAGE COMPARISONS. FOUR GOALS FOR FUTURE RESEARCH EMERGE FROM THE DISCUSSION: A MEASURE OF PROGRAM COMPREHENSION, AN EXAMINATION OF PARTICULAR SYNTACTIC FEATURES IN COMMON USE, AN ATTACK ON THE SPECIAL PROBLEMS OF LARGE-SCALE PROGRAMS, AND AN ATTEMPT TO ABSORB THE FINDINGS OF CONTEMPORARY PSYCHOLINGUISTICS. (A)
50P, 49R.

160 PROGRAMMING
GRIEM, P.D., JR. TOWARD A PROGRAMMING DISCIPLINE. DATAMATION, 1972, 18, 140.
DESCRIPTION:
THIS IS A FORUM LETTER THAT POINTS TO SOME OF THE REASONS THAT SOFTWARE IS GROWING MORE EXPENSIVE, SHODDY, AND DIFFICULT TO STANDARDIZE. THERE ARE MORE TOOLS, LANGUAGES, AND FEATURES, BUT NOT, IN GENERAL, BETTER ONES. THE STANDARDIZATION EFFORTS UNDERWAY MUST CONVERGE BEFORE REAL PROGRESS WILL BE POSSIBLE. PROJECTS ARE LATE, DO NOT MEET SPECIFICATIONS, AND TEND TO RE-INVENT THEIR COMPONENTS BECAUSE SCHEDULING IS NOT WELL UNDERSTOOD, SPECIFICATIONS ARE NOT REALISTIC AND STABLE, AND PROJECTS DO NOT BUDGET EXTRA TIME AND MONEY TO MAKE REUSABLE COMPONENTS AVAILABLE IN A FORM THAT WILL PERMIT REUSE. (GDC)
1P, 0R.

161 STRUCTURED PROGRAMMING
GRIES, D. ON STRUCTURED PROGRAMMING: A REPLY TO SMOLIAR. COMMUNICATIONS OF THE ACM, 1974, 17, 655-657.

162 PROGRAMMING
HALL, M.M., & KIDMAN, B.P. RUN MONITORING OF STUDENT PROGRAMMING. AUSTRALIAN COMPUTER JOURNAL, 1975, 7, 127-131.
DESCRIPTION:
THIS REPORT IS PRIMARILY CONCERNED WITH RESULTS OBTAINED FROM AN EXPERIMENTAL MONITORING OF COMPUTER RUNS MADE BY STUDENTS WHILE DEVELOPING ONE PARTICULAR BASIC PROGRAM. HIGH REPORTED ERROR RATES AND A WIDE VARIATION IN INDIVIDUAL RUN PATTERNS WERE FOUND, A LARGE NUMBER OF RUNS BEING REQUIRED BY MANY STUDENTS. THE NUMBER OF RUNS COULD NOT BE RELATED TO THE PROGRAM TEXT, BUT ONE PARAMETER WAS FOUND TO BE RELATED TO THE ACADEMIC GRADE OF THE STUDENTS. THE RESULTS OBTAINED SUGGEST THAT THOSE WHO RAN MOST PROGRAMS PER DAY DID NOT SAVE MAN-HOURS IN "DEBUGGING BY MACHINE." (A)
5P, 4R.

163 USER INVOLVEMENT IN SYSTEMS ANALYSIS
HALPERN, E.V. USER INVOLVEMENT IN THE SYSTEMS ANALYSIS FUNCTION. COMPUTER PERSONNEL, 1977, 6(1-2), 3-8.

164 NATURAL-LANGUAGE PROGRAMMING
HALPERN, M. FOUNDATIONS OF THE CASE FOR NATURAL-LANGUAGE PROGRAMMING. IEEE SPECTRUM, MARCH 1967, 4(3), 140-149.
DESCRIPTION:
THE PURPOSE OF THIS PAPER IS TO CLARIFY SOME OF THE MISCONCEPTIONS THAT IMPEDE USEFUL DISCUSSION OF THE QUESTION OF THE SUITABILITY OF NATURAL LANGUAGE FOR PROGRAMMING. IT IS ARGUED THAT: (1) NATURAL-LANGUAGE PROGRAMMING IS AN ATTEMPT TO PUT NONPROGRAMMERS IN A CLOSER RELATION WITH THE COMPUTER, (2) A NATURAL PROGRAMMING LANGUAGE MUST BE ABLE TO BE WRITTEN EASILY, NOT JUST READ EASILY, (3) PROCESSING NATURAL LANGUAGE IS QUALITATIVELY DIFFERENT FROM (AND EASIER THAN) TRANSLATING ONE LANGUAGE TO ANOTHER, (4) THE REDUNDANCY OF NATURAL LANGUAGE IS AN ADVANTAGE RATHER THAN A DISADVANTAGE, AND (5) NATURAL LANGUAGE PROGRAMMING WILL HELP BRIDGE THE MAN-MACHINE COMMUNICATION GAP. (MEA)
10P, 17R.

165 SOFTWARE PHYSICS
HALSTEAD, M.H. NATURAL LAWS CONTROLLING ALGORITHM STRUCTURE? SIGPLAN NOTICES, 1972, 7, 19-26 (ALSO: (TECHNICAL REPORT NO. TR66). LAFAYETTE, INDIANA: PURDUE UNIVERSITY, FEBRUARY 1972).

166 SOFTWARE PHYSICS
HALSTEAD, M.H. A THEORETICAL RELATIONSHIP BETWEEN MENTAL WORK AND MACHINE LANGUAGE PROGRAMMING (TECHNICAL REPORT NO. CSD-TR-67). LAFAYETTE, INDIANA: PURDUE UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, 1972.

167 SOFTWARE PHYSICS
HALSTEAD, M.H. AN EXPERIMENTAL DETERMINATION OF THE "PURITY" OF A TRIVIAL ALGORITHM. PERFORMANCE EVALUATION REVIEW (ACM SIGME), MARCH 1973, 2(1), 10-15 (ALSO: (TECHNICAL REPORT NO. CSD-TR-73). LAFAYETTE, INDIANA: PURDUE UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, 1973).
DESCRIPTION:
EIGHTEEN VERSIONS OF ONE ALGORITHM WERE GENERATED AND USED AS EXPERIMENTAL DATA TO STUDY "IMPURITIES" IN ALGORITHMS. THE RESULTS OF THE STUDY SHOWED THAT ANY VERSION WHICH EXHIBITS A) SELF-CANCELLING TERMS, B) EQUIVALENT OPERANDS, C) DUAL USAGE OF OPERANDS, OR D) UNFACTORED EXPRESSIONS, SHOULD NOT BE CONSIDERED A PURE ALGORITHMS. (O)
6P, 6R.

168 SOFTWARE PHYSICS
HALSTEAD, M.H. LANGUAGE LEVEL, A MISSING CONCEPT IN INFORMATION THEORY. PERFORMANCE EVALUATION REVIEW (ACM SIGME), MARCH 1973, 2(1), 7-9 (ALSO TECHNICAL REPORT NO. CSD-TR-75, LAFAYETTE, INDIANA: PURDUE UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, 1973).
DESCRIPTION:
THIS PAPER INTRODUCES THE CONCEPT OF LANGUAGE LEVEL, A QUANTITATIVE MEASURE RELATED TO THE AVERAGE POWER OF INDIVIDUAL STATEMENTS IN A PARTICULAR PROGRAMMING LANGUAGE. (O)
3P, 6R.

169 PROGRAMMING LANGUAGES
HALSTEAD, M.H. LANGUAGE SELECTION FOR APPLICATIONS. AFIPS CONFERENCE PROCEEDINGS, 1973, 42, 211-214.
DESCRIPTION:
IN CHOOSING A PROGRAMMING LANGUAGE WITH WHICH TO IMPLEMENT A GIVEN CLASS OF PROBLEMS, SEVERAL VARIABLES SHOULD BE CONSIDERED. IN GENERAL, HOWEVER, THE APPROACH USED INVOLVES ONLY ADVICE SUCH AS "FOR SCIENTIFIC AND ENGINEERING PROBLEMS, USE FORTRAN; FOR PROBLEMS WITH LARGE DATA BASES, USE COBOL; ETC." ALTHOUGH SUCH ADVICE MAY OCCASIONALLY BE CORRECT, IT MAY BE INCORRECT BECAUSE IT IGNORES MOST OF THE IMPORTANT VARIABLES INVOLVED IN LANGUAGE SELECTION. THIS PAPER EXAMINES THE DIMENSIONS ALONG WHICH PROGRAMMING LANGUAGES CAN BE COMPARED AND DISCUSSES THE RELATIVE IMPORTANCE OF EACH OF THESE DIMENSIONS IN LANGUAGE SELECTION. (MEA)
4P, 14R.

170 SOFTWARE PHYSICS
HALSTEAD, M.H. SOFTWARE PHYSICS: BASIC PRINCIPLES (TECHNICAL REPORT NO. RJ 1582). SAN JOSE, CALIFORNIA: IBM RESEARCH LABORATORY, 1975.

171 SOFTWARE PHYSICS
HALSTEAD, M.H. THE ESSENTIAL DESIGN CRITERION FOR COMPUTER LANGUAGES: SOFTWARE SCIENCE (TECHNICAL REPORT NO. CSD-TR-191). LAFAYETTE, INDIANA: PURDUE UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, 1976.
DESCRIPTION:
APPENDIX CONTAINS ELEVEN TEST PROGRAMS AND A DETAILED COUNT OF OPERATORS AND OPERANDS. (O)
30P, 0R.

172 SOFTWARE PHYSICS
HALSTEAD, M.H. ELEMENTS OF SOFTWARE SCIENCE. NEW YORK: ELSEVIER NORTH-HOLLAND, 1977.
DESCRIPTION:
THIS COMPREHENSIVE TEXT COLLECTS AND ORGANIZES TECHNOLOGICAL DEVELOPMENTS IN THE AREA OF SOFTWARE PHYSICS PREVIOUSLY UNAVAILABLE IN A SINGLE VOLUME. THE AUTHOR CLEARLY DEFINES TERMINOLOGY AND HAS ORGANIZED THE BOOK WITH EACH CHAPTER BUILDING ON THE MATERIAL PRECEDING IT. THEREFORE, THE TEXT WILL SERVE AS BOTH AN INTRODUCTION TO THE SCIENCE AS WELL AS A BASIS FOR CONTINUED DEVELOPMENT. (O)
141P, 56R.

173 SOFTWARE PHYSICS
HALSTEAD, M., & BAYER, R. ALGORITHM DYNAMICS. IN PROCEEDINGS, ACM NATIONAL CONFERENCE, ATLANTA, GA, AUGUST 1973. NEW YORK: ASSOCIATION FOR COMPUTING MACHINERY, 1973, 126-135.
DESCRIPTION:
A TECHNIQUE FOR MEASURING SIMPLE STRUCTURAL PROPERTIES OF ALGORITHMS IS DESCRIBED. USING THESE MEASURES, IT IS FOUND THAT FOR A NON-TRIVIAL CLASS OF ALGORITHMS THERE IS A QUANTITATIVE RELATIONSHIP BETWEEN OPERATORS AND OPERANDS AND THEIR USAGE. PROPERTIES OF "FULL" AND "REDUCED" ALGORITHMS ARE THEN EXPLORED, AND SHOWN TO PREDICT THE QUANTITATIVE RELATIONSHIP OBSERVED. (A)

174 SOFTWARE PHYSICS
HALSTEAD, M.H., GORDON, R.D., & ELSHOFF, J.L. ON SOFTWARE PHYSICS AND GM'S PL/1 PROGRAMS (RESEARCH PUBLICATION NO. GMR-2175). DETROIT, MICHIGAN: GENERAL MOTORS RESEARCH LABS, 1976.
DESCRIPTION:
A BETTER UNDERSTANDING OF BOTH HALSTEAD'S SOFTWARE PHYSICS AND GM'S PL/1 PROGRAMS HAS RESULTED FROM APPLYING THE THEORY OF SOFTWARE PHYSICS TO THE PROGRAMS. THE LARGE VOLUME OF DATA EXTRACTED FROM THE PROGRAMS HAS LED TO TWO SIGNIFICANT EXTENSIONS OF SOFTWARE PHYSICS. A GLOBAL LEVEL OF AN ALGORITHM WHICH DEPENDS ONLY UPON A LANGUAGE AND ITS USE HAS BEEN DERIVED. T DEVELOPMENT HAS LED TO A PREDICTIVE MEASURE FOR ESTIMATING THE TIME JIRED TO WRITE A PROGRAM. IN TURN, THESE MEASURES GIVE NEW INSIGHTS INTO .E QUALITY OF THE PROGRAMMING THAT IS BEING DONE. (A)
26P, 9R.

175 SOFTWARE PHYSICS
HALSTEAD, M.H., & ZISLIS, P.M. EXPERIMENTAL VERIFICATION OF TWO THEOREMS OF SOFTWARE PHYSICS (TECHNICAL REPORT NO. CSD-TR-97). LAFAYETTE, INDIANA: PURDUE UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, 1973.

176 SOFTWARE DEVELOPMENT
HANEY, F.M. MODULE CONNECTION ANALYSIS - A TOOL FOR SCHEDULING SOFTWARE DEBUGGING ACTIVITIES. AFIPS CONFERENCE PROCEEDINGS, 1972, 40, 173-179.
DESCRIPTION:
THIS PAPER DESCRIBES A SIMPLE MODEL FOR THE EFFECT OF 'RIPPLING CHANGES' IN A LARGE SYSTEM. THE MODEL CAN BE USED TO ESTIMATE THE NUMBER OF CHANGES AND A RELEASE STRATEGY FOR STABILIZING A SYSTEM GIVEN ANY SET OF INITIAL CHANGES. THE MODEL CAN BE CRITICIZED FOR BEING SIMPLISTIC, YET IT SEEMS TO DESCRIBE THE ESSENCE OF THE PROBLEM OF STABILIZING A SYSTEM. IT IS CLEAR, TO THE AUTHOR AT LEAST, THAT EXPERIMENTATION WITH THE MODULE CONNECTION MODEL COULD HAVE PREVENTED A SIGNIFICANT PORTION OF THE SCHEDULE DELAY THAT OCCURRED FOR MANY LARGE SYSTEMS. (A)
7P, 5R.

177 PROGRAM COMPLEXITY
HANSEN, W.J. MEASUREMENT OF PROGRAM COMPLEXITY BY THE PAIR (CYCLOMATIC NUMBER, OPERATOR COUNT). SIGPLAN NOTICES, MARCH 1978, 13(3), 29-33.
DESCRIPTION:
IN A RECENT PAPER, T.J. MCCABE (1976) INTRODUCED THE CYCLOMATIC NUMBER OF A PROGRAM'S FLOW GRAPH AS A MEASURE OF ITS COMPLEXITY. G.J. MYERS (1977) PROPOSED AN IMPROVED MEASURE CONSISTING OF AN INTERVAL WITH THE ORIGINAL MEASURE AS ITS UPPERBOUND. I WILL ARGUE BELOW THAT -- IF TWO VALUES ARE TO BE PRESENTED AS A MEASURE -- IT IS PREFERABLE TO COUPLE A VARIATION OF THE CYCLOMATIC NUMBER WITH A MEASURE OF THE PROGRAM'S EXPRESSION COMPLEXITY. (A)
5P, 5R.

178 STATISTICS ON USER BEHAVIOR IN TIME-SHARING SYSTEM
HARALAMBOPOULOS, G., & NAGY, G. PROFILE OF A UNIVERSITY COMPUTER USER COMMUNITY. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1977, 9, 287-313.
DESCRIPTION:
THE DATA RECORDED OVER A ONE-YEAR PERIOD ON THE PERMANENTLY MOUNTED SYSTEM MONITOR TAPE AT THE UNIVERSITY OF NEBRASKA IS ANALYZED WITH A VIEW TO DETERMINE PROMINENT COMPUTER USER CHARACTERISTICS. THE USERS ARE DIVIDED INTO DISTINCT CATEGORIES ACCORDING TO THEIR PATTERNS OF COMPUTER RESOURCE UTILIZATION. (A)
27P, 49R.

179 PROGRAMMING LANGUAGES

HARDGRAVE, W.T. POSITIONAL VERSUS KEYWORD PARAMETER COMMUNICATION IN PROGRAMMING LANGUAGES. SIGPLAN NOTICES, MAY 1976, 11 (5), 52-58.

DESCRIPTION:

IN RECENT YEARS, THE STUDY OF PROGRAMMING LANGUAGES (E.G. DIJKSTRA, DAHL, ZAHN, AND KNUTH) HAS BEEN LARGELY DEVOTED TO THE DESIGN OF IMPROVED CONTROL STRUCTURES; AND WHILE THESE STUDIES HAVE PRODUCED SIGNIFICANT RESULTS IN IMPROVING CODE, THERE ARE A NUMBER OF OTHER AREAS THAT FALL UNDER THE GENERAL HEADING OF "STRUCTURED PROGRAMMING" IN WHICH SIGNIFICANT RESULTS ARE SO FAR LACKING. ONE OF THESE IS THE PROBLEM OF PARAMETRIC COMMUNICATION BETWEEN PROCEDURES. THE TRADITIONAL "POSITIONAL" APPROACH AND THE NEWER "KEYWORD" APPROACH ARE BRIEFLY REVIEWED, EXAMPLES OF PROBLEM AREAS ENCOUNTERED WHEN USING THE POSITIONAL APPROACH ARE CITED, AND FINALLY THE KEYWORD APPROACH IS SUGGESTED AS A VIABLE AUGMENTATION. (A)
7P, 6R.

180 SPECIFICATIONS

HARTMAN, P.H., & OWENS, D.H. HOW TO WRITE SOFTWARE SPECIFICATIONS. IN PROCEEDINGS, FALL JOINT COMPUTER CONFERENCE, 1967, 779-790.

DESCRIPTION:

IN GENERAL, COMPUTER SOFTWARE IS GETTING MORE COMPLICATED AT AN INCREASING RATE. UNFORTUNATELY, THE PEOPLE WHO HAVE TO DEVELOP THIS SOFTWARE HAVE BEEN AT IT FOR ONLY A RELATIVELY SHORT TIME. THE RESULT IS PREDICTABLE: EVER-LARGER SOFTWARE TROUBLES. THIS, IN TURN, LEADS MANY PEOPLE TO FEEL INSECURE ABOUT SOFTWARE DEVELOPMENT, TO THINK OF IT AS A MODERN "BLACK ART" FOR WHICH EVEN THE MOST ABLE PRACTITIONERS LOSE THE RECIPE EVERY FEW MONTHS.

WE THINK MANY OF THESE TROUBLES ARE SELF-INFLICTED, THE RESULT OF A NATURAL DESIRE TO GET ON THE COMPUTER (AFTER ALL, ISN'T THAT WHAT PROGRAMMING IS ALL ABOUT?) INSTEAD OF "WASTING" TIME IN PLANNING AND OTHER "PAPER SHUFFLING."

TO ILLUSTRATE THIS POINT, WE PARTICULARLY DISCUSS THE SPECIFYING OF A FORTRAN COMPILER.

THIS PAPER IS NOT A "COOKBOOK" ON HOW TO WRITE A COMPILER (THAT PROBLEM IS TOO BROAD FOR A BRIEF DISCUSSION), NOR EVEN WHAT KIND OF COMPILER TO WRITE. INSTEAD, IT IS AN INTRODUCTION TO A WAY OF THINKING ABOUT AND DEFINING THE PRODUCT TO BE DEVELOPED. (A, ABBR.)
12P, 0R.

181 PERSONALIZED MAN-COMPUTER DIALOGUE

HEAFNER, J.F. A METHODOLOGY FOR SELECTING AND REFINING MAN-COMPUTER LANGUAGES TO IMPROVE USERS' PERFORMANCE (REPORT NO. ISI/RR-74-21). MARINA DEL RAY, CALIFORNIA: UNIVERSITY OF SOUTHERN CALIFORNIA, INFORMATION SCIENCES INSTITUTE, SEPTEMBER 1974. (NTIS NO. AD 787684)

DESCRIPTION:

THIS REPORT DESCRIBES A METHODOLOGY (SUPPORTED BY A SOFTWARE PACKAGE) TO MODEL, MEASURE, ANALYZE, AND EVALUATE USERS' PERFORMANCE IN A MESSAGE COMMUNICATION SYSTEM ENVIRONMENT. THE THESES OF THE REPORT ARE: (1) THAT MODELS OF USERS AND SERVICES CAN ACCURATELY BE USED AS PREDICTORS IN SELECTING A LANGUAGE FORM, FOR AN APPLICATION, WHICH WILL RESULT IN HIGH USERS' PERFORMANCE, AND (2) THAT SUCH A LANGUAGE FORM IS ONLY AN APPROXIMATION (IN TERMS OF YIELDING OPTIMAL USER'S PERFORMANCE) DUE TO WITHIN VARIANCES OF USER AND SERVICE-CLASSES, HENCE INDIVIDUAL, ON-LINE REGULATION OF LANGUAGE CONSTRUCTS IS NECESSARY TO FURTHER IMPROVE PERFORMANCE.

THIS REPORT DEVELOPS APPROPRIATE MODELS AND ALGORITHMS, AND STATES HYPOTHESES RELATING THE INTERACTIVE EFFECTS OF USERS, SERVICES, LANGUAGE FORMS, AND OTHER VARIABLES IMPORTANT IN MAN-MACHINE DISCOURSE. AN EXPERIMENTAL DESIGN IS PRESENTED, WHICH TESTS THE MAJOR HYPOTHESES. (A)
64P, 10R.

182 PROTOCOL ANALYSIS FOR DESIGN OF MAN-COMPUTER DIALOGUE

HEAFNER, J.F. PROTOCOL ANALYSIS OF MAN-COMPUTER LANGUAGES: DESIGN AND PRELIMINARY FINDINGS (REPORT NO. ISI/RR-75-34). MARINA DEL REY, CALIFORNIA: UNIVERSITY OF SOUTHERN CALIFORNIA, INFORMATION SCIENCES INSTITUTE, JULY 1975. (NTIS NO. AD A 13568)

DESCRIPTION:

THIS REPORT DESCRIBES AN ON-GOING STUDY IN MAN-MACHINE COMMUNICATIONS. THE STUDY'S MAIN PREMISE IS THAT IN DEVELOPING MAN-COMPUTER LANGUAGES, ONE SHOULD CONSIDER THE USERS' NEEDS AND HABITS, AS WELL AS FEATURES OF THE COMPUTER SERVICE. THE PROBLEM IN DOING SO IS THAT THE DESIGNER DOES NOT HAVE SUFFICIENT QUANTITATIVE INFORMATION ABOUT THE USERS TO ENABLE HIM TO SPECIFY LANGUAGES PERMITTING NEAR-OPTIMAL PERFORMANCE. THE STUDY PROPOSES AND TESTS A METHOD TO ACHIEVE A CLOSER FIT BETWEEN USERS AND THEIR COMPUTER LANGUAGES BY INVOLVING POTENTIAL USERS IN THE DESIGN PROCESS.

TOKEN LANGUAGES OF SEVERAL SYNTACTIC FORMS ARE DEFINED. THEN, RESEARCH HYPOTHESES ARE STATED CONCERNING THE USERS' PREFERENCES REGARDING THE LANGUAGE STRUCTURE AND VOCABULARY. NEXT, AN EXPERIMENT DESIGN IS DESCRIBED, BASED ON A STATISTICAL MODEL OF OBSERVATIONS OF COMMANDS ENTERED BY USERS AS THEY PERFORM A STANDARDIZED TASK. THE METHOD IS TESTED BY PROTOCOL ANALYSIS WITH SUBJECTS WHO ARE POTENTIAL USERS. IN THE PROTOCOL ANALYSIS, SUBJECTS VOCALLY STATED COMMANDS IN EACH OF THE TOKEN LANGUAGES AS THEY PERFORMED THE STANDARDIZED TASK. THESE RESPONDENTS WERE REQUESTED TO CHANGE THE GRAMMAR OF EACH LANGUAGE (DURING THE TASK) TO MAKE IT MORE NATURAL FOR THEM TO USE. THEIR TASK INPUTS WERE USED TO TEST THE HYPOTHESIS. THE REPORT CONCLUDES THAT THE METHOD OF MODELLING USERS AND THEN TESTING DRAFT LANGUAGES IS USEFUL IN LANGUAGE DESIGN, SINCE THERE WAS A CONSENSUS OF USERS' OPINIONS AS TO SPECIFIC LANGUAGE IMPROVEMENTS. (A)
227P, 24R.

183 PROGRAMMING LANGUAGES

HEHNER, E.C.R. MERLIN: TOWARDS AN IDEAL PROGRAMMING LANGUAGE (TECHNICAL REPORT NO. CSRG-57). TORONTO, CANADA: UNIVERSITY OF TORONTO, COMPUTER SYSTEMS RESEARCH GROUP, 1975.

DESCRIPTION:

ACCEPTING THE PREMISE THAT MACHINES SHOULD BE DESIGNED TO SUIT THE LANGUAGES THAT WILL BE IMPLEMENTED ON THEM, AND NOT VICE VERSA, THIS PROJECT EXPLORES ISSUES OF LANGUAGE DESIGN NOT INFLUENCED BY IMPLEMENTATION CONSIDERATIONS. OUR CONCERN IS ONLY CONVENIENCE FOR THE DEVELOPMENT AND CLEAR EXPRESSION OF ALGORITHMS. UNDER THIS, WE INCLUDE SIMPLICITY, A MINIMUM OF TERMINOLOGY, FREEDOM FROM USELESS RESTRICTIONS, LACK OF USELESS REDUNDANCY, INCLUSION OF USEFUL REDUNDANCY, AND AFFINITY OF LANGUAGE STRUCTURES TO THOUGHT STRUCTURES. AN ATTEMPT IS MADE TO INTRODUCE A DISCIPLINE OF LANGUAGE DESIGN, RATHER THAN STICKING TOGETHER FAVORITE LANGUAGE FEATURES. (A)
83P, 16R.

184 NATURAL LANGUAGE PROGRAMMING

HEIDORN, G.E. AUTOMATIC PROGRAMMING THROUGH NATURAL LANGUAGE DIALOGUE: A SURVEY. IBM JOURNAL OF RESEARCH AND DEVELOPMENT, 1976, 20, 302-313.

DESCRIPTION:

THIS PAPER DESCRIBES AND COMPARES FOUR RESEARCH PROJECTS WHOSE GOAL IS TO DEVELOP AN AUTOMATIC PROGRAMMING SYSTEM THAT CAN CARRY ON A NATURAL LANGUAGE DIALOGUE WITH A USER ABOUT HIS REQUIREMENTS AND THEN PRODUCE AN APPROPRIATE PROGRAM. IT ALSO DISCUSSES SOME OF THE IMPORTANT ISSUES IN THIS RESEARCH AREA. (A)
12P, 39R.

185 PROGRAMMING

HENDERSON, P., & SNOWDON, R. AN EXPERIMENT IN STRUCTURED PROGRAMMING. BIT, 1972, 12, 38-53.

DESCRIPTION:

THE CONSTRUCTION OF A PROGRAM TO SOLVE A SIMPLE PROBLEM, WRITTEN USING A TOP-DOWN STRUCTURAL APPROACH, IS DESCRIBED. AN INDEPENDENT ANALYSIS OF THIS PROGRAM IS PROVIDED COMMENTING ON THE POSSIBLE PROBLEMS THAT ARISE FROM THE USE OF SUCH A TECHNIQUE. (A)

16P, 5R.

186 SOFTWARE ENGINEERING

HEWITT, C.E., & SMITH, B. TOWARDS A PROGRAMMING APPRENTICE. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1(2), 26-45 (ALSO TECHNICAL REPORT, CAMBRIDGE, MASSACHUSETTS: MASSACHUSETTS INSTITUTE OF TECHNOLOGY, ARTIFICIAL INTELLIGENCE LABORATORY, 1975). (NTIS NO. AD A009519)

DESCRIPTION:

THE PLANNER PROJECT IS CONSTRUCTING A PROGRAMMING APPRENTICE TO ASSIST IN KNOWLEDGE BASED PROGRAMMING. WE WOULD LIKE TO PROVIDE AN ENVIRONMENT WHICH HAS SUBSTANTIAL KNOWLEDGE OF THE SEMANTIC DOMAIN FOR WHICH THE PROGRAMS ARE BEING WRITTEN, AND KNOWLEDGE OF THE PURPOSES THAT THE PROGRAMS ARE SUPPOSED TO SATISFY. FURTHER, WE WOULD LIKE TO MAKE IT EASY FOR THE PROGRAMMER TO COMMUNICATE THE KNOWLEDGE ABOUT THE PROGRAM TO THE APPRENTICE. THE APPRENTICE IS TO AID EXPERT PROGRAMMERS IN THE FOLLOWING KINDS OF ACTIVITIES:

1) ESTABLISHING AND MAINTAINING CONSISTENCY OF SPECIFICATIONS,
2) VALIDATING THAT MODULES MEET THEIR SPECIFICATIONS,
3) ANSWERING QUESTIONS ABOUT DEPENDENCIES BETWEEN MODULES,
4) ANALYZING IMPLICATIONS OF PERTURBATIONS IN MODULES,
5) ANALYZING IMPLICATIONS OF PERTURBATIONS IN SPECIFICATIONS.

WE USE CONTRACTS (PROCEDURAL SPECIFICATIONS) TO EXPRESS WHAT IS SUPPOSED TO BE ACCOMPLISHED AS OPPOSED TO HOW IT IS SUPPOSED TO BE DONE. THE IDEA IS THAT AT LEAST TWO PROCEDURES SHOULD BE WRITTEN FOR EACH MODULE IN A SYSTEM. ONE PROCEDURE IMPLEMENTS A METHOD FOR ACCOMPLISHING A DESIRED TRANSFORMATION AND THE OTHER CAN CHECK THAT THE TRANSFORMATION HAS IN FACT BEEN ACCOMPLISHED. THE PROGRAMMING APPRENTICE IS DESIGNED FOR INTERACTIVE USE BY EXPERT PROGRAMMERS IN THE META-EVALUATION OF IMPLEMENTATIONS IN THE CONTEXT OF THEIR CONTRACTS AND BACKGROUND KNOWLEDGE. META-EVALUATION PRODUCES JUSTIFICATION WHICH MAKES EXPLICIT EXACTLY HOW THE MODULE DEPENDS ON THE CONTRACTS OF OTHER MODULES AND ON THE BACKGROUND KNOWLEDGE. THE JUSTIFICATION IS USED IN ANSWERING QUESTIONS ON THE BEHAVIOR DEPENDENCIES BETWEEN MODULES AND IN ANALYZING THE IMPLICATIONS ON PERTURBATIONS IN SPECIFICATIONS AND/OR IMPLEMENTATION. (A)

20P, 62R.

187 NATURAL-LANGUAGE PROGRAMMING

HILL, I.D. WOULDN'T IT BE NICE IF WE COULD WRITE COMPUTER PROGRAMS IN ORDINARY ENGLISH -- OR WOULD IT? COMPUTER BULLETIN, JUNE 1972, 16(6), 306-312.

DESCRIPTION:

ONE ARGUMENT THAT IS FREQUENTLY MADE IN FAVOR OF NATURAL-LANGUAGE PROGRAMMING IS THAT PEOPLE SHOULD BE ABLE TO COMMUNICATE WITH COMPUTERS IN THE SAME WAY THAT THEY COMMUNICATE WITH EACH OTHER. WHILE IT IS DESIRABLE TO HAVE A COMMON MODE OF COMMUNICATION, THIS DOES NOT IMPLY THAT WE NEED TO TEACH COMPUTERS ENGLISH; AN ALTERNATIVE IS TO TEACH PEOPLE TO COMMUNICATE WITH EACH OTHER THROUGH UNAMBIGUOUS INSTRUCTIONS. THIS PAPER WILL CONSIDER THE INTRICACIES IN NATURAL ENGLISH THAT PROHIBIT NATURAL LANGUAGE AND SIMULTANEOUSLY ILLUSTRATE THE NEED FOR PEOPLE TO USE PROGRAMMING LANGUAGES IN THEIR INTERACTIONS WITH OTHERS. (MEA)

7P, 12R.

188 PROGRAMMING LANGUAGES

HOARE, C.A.R. HINTS FOR PROGRAMMING LANGUAGE DESIGN (TECHNICAL REPORT NO. CS-73-403). STANFORD, CALIFORNIA: STANFORD UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, DECEMBER 1973. (NTIS NO. AD 773391)

DESCRIPTION:

THIS PAPER (BASED ON A KEYNOTE ADDRESS PRESENTED AT THE SIGACT/SIGPLAN SYMPOSIUM ON PRINCIPLES OF PROGRAMMING LANGUAGE, BOSTON, OCTOBER 1-3, 1973) PRESENTS THE VIEW THAT A PROGRAMMING LANGUAGE IS A TOOL WHICH SHOULD ASSIST THE PROGRAMMER IN THE MOST DIFFICULT ASPECTS OF HIS ART, NAMELY PROGRAM DESIGN, DOCUMENTATION, AND DEBUGGING. IT DISCUSSES THE OBJECTIVE CRITERIA FOR EVALUATING A LANGUAGE DESIGN, AND ILLUSTRATES THEM BY APPLICATION TO LANGUAGE FEATURES OF BOTH HIGH LEVEL LANGUAGES AND MACHINE CODE PROGRAMMING. IT CONCLUDES WITH AN ANNOTATED READING LIST, RECOMMENDED TO ALL INTENDING LANGUAGE DESIGNERS. (A)

31P, 9R.

189 NATURAL-LANGUAGE PROGRAMMING

HOBBS, J.R. WHAT THE NATURE OF NATURAL LANGUAGE TELLS US ABOUT HOW TO MAKE NATURAL-LANGUAGE-LIKE PROGRAMMING LANGUAGES MORE NATURAL. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 85-93 (ALSO: SIGART NEWSLETTER, AUGUST 1977, 64, 85-93).

DESCRIPTION:

WHEN A STUDENT IS LEARNING AN ALGORITHM FROM A TEXTBOOK, HIS FIRST APPROACH IS FREQUENTLY THROUGH AN ENGLISH DESCRIPTION. THIS IS NORMALLY EASIER TO UNDERSTAND THAN RAW CODE, AND SOMETIMES EASIER THAN A FLOWCHART, IN SPITE OF THE FACT THAT PROGRAMMING LANGUAGES ARE DESIGNED FOR ALGORITHM SPECIFICATION WHILE ENGLISH IS ONLY PRESSED INTO ITS SERVICE. IF THE ENGLISH IS EASIER TO UNDERSTAND, IT IS LIKELY THAT IT HAS MANY FEATURES THAT WOULD EASE PROGRAMMING ITSELF. THIS PAPER INVESTIGATES SOME OF THESE FEATURES. (A)

9P, 9R.

190 PROGRAMMING

HOC, J.M. THE ROLE OF MENTAL REPRESENTATION IN LEARNING A PROGRAMMING LANGUAGE. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1977, 9, 87-105.

DESCRIPTION:

A THEORETICAL FRAMEWORK HAS BEEN DEFINED AND BEGUN TO GET A VALIDATION BY A PRELIMINARY EXPERIMENT. THIS EXPERIMENT ENABLED US TO SHOW THAT THE PROGRAMMING LANGUAGE IS INTERIORIZED BY THE SUBJECT STEP BY STEP IN THE FORM OF A "SYSTEME DE REPRESENTATION ET DE TRAITEMENT" (SRT) IN WHICH THE SKILLED PROGRAMMER WILL BE ABLE TO ANALYZE THE PROBLEMS. BEFORE THIS, HE PERFORMS HIS ANALYSIS IN OTHER SRTS MORE OR LESS COMPATIBLE WITH THE PROGRAMMING LANGUAGE. NINETEEN SUBJECTS, AT VARIOUS LEVELS OF TRAINING, HAD TO CONSTRUCT THE COBOL-ALGORIGRAM OF THE PROBLEM OF CONTROLLING A SUBWAY TICKET MACHINE. AN ANALYSIS OF THE ERRORS AND A DESCRIPTION OF ANALYSIS STRATEGIES, BY MEANS OF 22 VARIABLES, HAVE BEEN PERFORMED IN ORDER TO SPECIFY THREE IMPORTANT STEPS THROUGH THE PROGRAMMING LANGUAGE LEARNING. (A)

19P, 20R.

191 PROGRAMMING
HOLTON, J.B. ARE THE NEW PROGRAMMING TECHNIQUES BEING USED? DATAMATION, JULY 1977, 23(7), 97-103.
DESCRIPTION:
IT IS OFTEN CLAIMED THAT THE USE OF EFFECTIVE SYSTEM DEVELOPMENT TECHNIQUES CAN LEAD TO GREATER SUCCESS IN BUSINESS DATA PROCESSING FUNCTIONS. THIS PAPER PRESENTS THE RESULTS OF A SURVEY DIRECTED AT DETERMINING HOW WIDESPREAD IS THE USE OF SUCH TECHNIQUES AND HOW EFFECTIVE THEY ARE. THE FOLLOWING TECHNIQUES WERE INVESTIGATED: STRUCTURED PROGRAMMING, TOP-DOWN DESIGN AND IMPLEMENTATION, THE STRUCTURED WALK-THROUGH, PROGRAMMER TEAM OPERATIONS, AND USE OF A PROGRAM DEVELOPMENT SUPPORT LIBRARY. IT APPEARS THAT THESE TECHNIQUES ARE NOT WIDELY USED, BUT THEY HAVE BEEN FOUND TO BE VERY USEFUL WHEN THEY HAVE BEEN TRIED. (MEA)
5P, 0R.

192 PROGRAMMING
HORNING, J.J., & WORTMAN, D.B. SOFTWARE HUT: A COMPUTER PROGRAM ENGINEERING PROJECT IN THE FORM OF A GAME. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1977, SE-3, 325-330.
DESCRIPTION:
THE SOFTWARE HUT (A SMALL SOFTWARE HOUSE) IS A COURSE PROJECT DESIGNED FOR A GRADUATE-LEVEL COURSE IN COMPUTER PROGRAM ENGINEERING. THIS PAPER DESCRIBES THE SOFTWARE HUT PROJECT AND DISCUSSES THE AUTHORS' EXPERIENCE USING IT IN GRADUATE COURSES AT THE UNIVERSITY OF TORONTO. SUGGESTIONS FOR IMPROVEMENTS IN THE PROJECT ARE GIVEN. (A)
6P, 9R.

193 STRUCTURED PROGRAMMING
HOROWITZ, E. FORTRAN: CAN IT BE STRUCTURED -- SHOULD IT BE? COMPUTER, JUNE 1975, 8(6), 30-37.
DESCRIPTION:
IN ADDITION TO THE COMMONLY ACCEPTED CRITERIA OF CORRECTNESS AND EFFICIENCY, STRUCTURED PROGRAMMING EMPHASIZES THREE CRITERIA FOR DESIGN AND CODING -- READABILITY, MODIFIABILITY, AND PROVABILITY. THIS PAPER EXPLORES THE ADVANTAGES AND DISADVANTAGES OF STRUCTURED FORTRAN. IT IS CONCLUDED THAT STRUCTURED FORTRAN HAS BOTH GOOD POINTS AND LIMITATIONS, THAT USING STRUCTURED FORTRAN DOES NOT NECESSARILY IMPLY THE USE OF STRUCTURED PROGRAMMING, BUT ALSO THAT IT HAS SEVERAL ADVANTAGES THAT SHOULD PROVE BENEFICIAL WHEN IT IS CAREFULLY APPLIED. (MEA)
8P, 10R.

194 LARGE SOFTWARE SYSTEMS
HOROWITZ, E. (ED.) PRACTICAL STRATEGIES FOR DEVELOPING LARGE SOFTWARE SYSTEMS. READING, MASSACHUSETTS: ADDISON-WESLEY, 1975.

195 COMPUTER PERSONNEL
HUNT, D., & RANDHAWA, B.S. RELATIONSHIP BETWEEN AND AMONG COGNITIVE VARIABLES AND ACHIEVEMENT IN COMPUTATIONAL SCIENCE. EDUCATIONAL AND PSYCHOLOGICAL MEASUREMENT, 1973, 33, 921-928.
DESCRIPTION:
A STUDY WAS CONDUCTED TO TEST THE VALIDITY OF AN INTUITIVE ANALYSIS OF THE REQUISITE COGNITIVE ABILITIES REQUIRED FOR SUCCESS IN A SECOND YEAR COMPUTER SCIENCE COURSE. TWELVE EXISTING TESTS OF COGNITIVE ABILITIES WERE UTILIZED. FOUR OF THESE TESTS WERE FOUND TO BE USEFUL PREDICTORS OF SUCCESS, AS MEASURED BY COURSE GRADE. (MEA)
8P, 5R.

196 USER PROFILE
HUNT, E., DIEHR, G., & GARNATZ, D. WHO ARE THE USERS? AN ANALYSIS OF COMPUTER USE IN A UNIVERSITY COMPUTER CENTER. AFIPS CONFERENCE PROCEEDINGS, 1971, 38, 231-238 (ALSO TECHNICAL REPORT NO. 70-09-05, SEATTLE, WASHINGTON: UNIVERSITY OF WASHINGTON, COMPUTER SERVICE GROUP, SEPTEMBER 1970).

197 SOFTWARE DEVELOPMENT AND MAINTENANCE
IVIE, E.L. THE PROGRAMMER'S WORKBENCH -- A MACHINE FOR SOFTWARE DEVELOPMENT. COMMUNICATIONS OF THE ACM, 1977, 20, 746-753.
DESCRIPTION:
ON ALMOST ALL SOFTWARE DEVELOPMENT PROJECTS THE ASSUMPTION IS MADE THAT THE PROGRAM DEVELOPMENT FUNCTION WILL BE DONE ON THE SAME MACHINE ON WHICH THE EVENTUAL SYSTEM WILL RUN. IT IS ONLY WHEN THIS PRODUCTION MACHINE IS UNAVAILABLE OR WHEN ITS PROGRAMMING ENVIRONMENT IS TOTALLY INADEQUATE THAT ALTERNATIVES ARE CONSIDERED. IN THIS PAPER IT IS SUGGESTED THAT THERE ARE MANY OTHER SITUATIONS WHERE IT WOULD BE ADVANTAGEOUS TO SEPARATE THE PROGRAM DEVELOPMENT AND MAINTENANCE FUNCTIONS ONTO A SPECIALIZED COMPUTER WHICH IS DEDICATED TO THAT PURPOSE. SUCH A COMPUTER IS HERE CALLED A PROGRAMMER'S WORKBENCH. THE FOUR BASIC SECTIONS OF THE PAPER INTRODUCE THE SUBJECT, OUTLINE THE GENERAL CONCEPT, DISCUSS AREAS WHERE SUCH AN APPROACH MAY PROVE BENEFICIAL, AND DESCRIBE AN OPERATIONAL SYSTEM UTILIZING THIS CONCEPT. (A) 8P, 7R.

198 PROGRAMMING
JACKSON, K., & BUCHAN, D.E. AN EXERCISE IN PROGRAM DESIGN (REPORT NO. RRE-MEMO-2710). MALVERN, ENGLAND: ROYAL RADAR ESTABLISHMENT, OCTOBER 1971.

199 PROGRAMMING
JACKSON, M.A. PRINCIPLES OF PROGRAM DESIGN. NEW YORK: ACADEMIC PRESS, 1975.

200 PROGRAMMER PRODUCTIVITY
JOHNSON, J.R. A WORKING MEASURE OF PRODUCTIVITY. DATAMATION, FEBRUARY 1977, 23(2), PP. 106-107; 109; 112.

201 PROGRAMMING
JOINT LOGISTICS COMMANDERS. FINAL REPORT OF THE JOINT LOGISTICS COMMANDERS ELECTRONIC SYSTEMS RELIABILITY WORKSHOP, 5-9 MAY 1975, AIRLIE HOUSE. WARRENTON, VIRGINIA: JOINT TECHNICAL COORDINATING GROUP ON ELECTRONIC EQUIPMENT RELIABILITY, DEPARTMENT OF THE ARMY, THE NAVY AND THE AIR FORCE, OCTOBER 1975.
DESCRIPTION:
THE DEPARTMENT OF DEFENSE IS CURRENTLY SEEKING METHODS TO IMPROVE THE FIELD RELIABILITY OF ITS ELECTRONIC SYSTEMS, PARTICULARLY WEAPON SYSTEMS. SINCE MODERN WEAPON SYSTEMS NOW CONTAIN AN INCREASING NUMBER OF EMBEDDED DIGITAL COMPUTER SUBSYSTEMS, THE QUESTION OF HOW TO DEVELOP HIGHLY RELIABLE COMPUTER PROGRAMS (OR SOFTWARE) HAS BECOME OF VITAL INTEREST TO EXECUTIVES AT THE HIGHEST LEVELS OF GOVERNMENT. THIS PAPER ADDRESSES THE SOFTWARE RELIABILITY QUESTION IN THIS CONTEXT AND DESCRIBES SOME OF THE PROBLEMS BEING ENCOUNTERED, THE REASONS FOR THEM, AND SUGGESTIONS FOR THEIR RESOLUTION. (A)

202 SPECIFICATIONS

JONES, M.N. HIPO FOR DEVELOPING SPECIFICATIONS. DATAMATION, MARCH 1976, 22(3), PP. 112; 114; 121; 125.

DESCRIPTION:

A PRECISE DEFINITION OF USER REQUIREMENTS IS ESSENTIAL TO THE DEVELOPMENT OF A CORRECT DATA PROCESSING SYSTEM. TOO FREQUENTLY, THE CONTENTS OF SPECIFICATION PACKAGES ARE FUZZY, INACCURATE, AND INCOMPLETE, RESULTING IN SYSTEMS WITH THESE SAME CHARACTERISTICS. IMAGINE, THEN, UNDERTAKING A DATA PROCESSING PROJECT WITH NO WRITTEN USER SPECIFICATIONS, YET DELIVERING A QUALITY SYSTEM WHICH MEETS USER REQUIREMENTS IN ALL RESPECTS. THIS MAY SEEM IMPOSSIBLE, BUT IT HAS BEEN ACHIEVED USING A TECHNIQUE KNOWN AS HIPO TO DEVELOP AND DOCUMENT SYSTEM SPECIFICATIONS. (A)

4P, 0R.

203 DOCUMENTATION

JUDD, D.R. THE DOCUMENTATION OF COMPUTER PROGRAMS. IN INFOTECH INFORMATION LTD. SOFTWARE ENGINEERING. BERKSHIRE, ENGLAND: INFOTECH INFORMATION LTD., 1972, 411-424.

DESCRIPTION:

THIS PAPER PROVIDES A BRIEF INTRODUCTION TO THE PURPOSES OF COMPUTER PROGRAM DOCUMENTATION. SEVERAL STAGES OF DOCUMENTATION ARE IDENTIFIED AND THE FUNCTIONS THAT EACH MUST FULFILL ARE SPECIFIED. THE STAGES OF DOCUMENTATION CONSIDERED ARE: PROGRAM DESIGN, ERROR DIAGNOSIS, OPERATIONS, MAINTENANCE, DEVELOPMENT, AND MARKETING. (MEA)

12P, 0R.

204 SOFTWARE TESTING AND VALIDATION

KANE, J.R., & YAU, S.S. CONCURRENT SOFTWARE FAULT DETECTION. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 87-99.

DESCRIPTION:

A MODULE IS AN ABSTRACT COMPONENT OF A SOFTWARE SYSTEM. IT MAY BE INTERPRETED AS A MACHINE INSTRUCTION, HIGH LEVEL LANGUAGE STATEMENT, SUBROUTINE, PROCEDURE, ETC. A SEQUENCE OF MODULES IS EXECUTED FOR EACH TRANSACTION PROCESSED BY THE SYSTEM. CONTROL FAULTS MANIFEST THEMSELVES AS INCORRECT EXECUTION SEQUENCES. A GRAPH-THEORETIC MODEL FOR SOFTWARE SYSTEMS IS PRESENTED WHICH PERMITS A SYSTEM TO BE CHARACTERIZED BY ITS SET OF ALLOWABLE EXECUTION SEQUENCES. IT IS SHOWN HOW A SYSTEM CAN BE STRUCTURED SO THAT EVERY EXECUTION SEQUENCE AFFECTED BY A CONTROL FAULT IS OBVIOUSLY IN ERROR, I.E., NOT IN THE ALLOWABLE SET DEFINED BY THE SYSTEM MODEL. FAULTS ARE DETECTED BY MONITORING THE EXECUTION SEQUENCE OF EVERY TRANSACTION PROCESSED BY THE SYSTEM AND COMPARING ITS EXECUTION SEQUENCE TO THE SET OF ALLOWABLE SEQUENCES. ALGORITHMS ARE PRESENTED BOTH FOR STRUCTURING A SYSTEM SO THAT ALL FAULTS CAN BE DETECTED AND FOR FAULT DETECTION CONCURRENT WITH SYSTEM OPERATION. SIMULATION RESULTS ARE PRESENTED WHICH SUPPORT THE THEORETICAL DEVELOPMENT OF THIS PAPER. (A)

13P, 10R.

205 AUTOMATIC PROGRAMMING
KANT, E. THE SELECTION OF EFFICIENT IMPLEMENTATION FOR A HIGH-LEVEL LANGUAGE. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 140-146 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 140-146).
DESCRIPTION:
THIS PAPER CONSIDERS THE PROBLEM OF IDENTIFYING AN EFFICIENT SET OF IMPLEMENTATIONS FOR THE ABSTRACT CONSTRUCTS IN A VERY HIGH LEVEL PROGRAM DESCRIPTION. LIBRA IS A SYSTEM THAT PRUNES AND EXPANDS A TREE OF PARTIALLY IMPLEMENTED PROGRAM DESCRIPTIONS, GIVEN A SET OF REFINEMENT RULES FOR GENERATING THE TREE. SEVERAL SETS OF RULES GROUP, ORDER, AND SELECT REFINEMENTS. THE ANALYSIS OF THE COST OF A PROGRAM (OR PROGRAM PART) AT ANY LEVEL OF REFINEMENT IS MAINTAINED FOR COST COMPARISONS BETWEEN DIFFERENT REFINEMENTS, FOR BOTTLENECK IDENTIFICATION, AND FOR BRANCH AND BOUND SEARCH. (A)
7P, 11R.

206 STRUCTURED PROGRAMMING
KATKUS, G.R. APPLYING STRUCTURED PROGRAMMING TO COMMAND, CONTROL, AND COMMUNICATION SOFTWARE DEVELOPMENT. COMPUTER MAGAZINE, JUNE 1975, 8(6), 43-47.
DESCRIPTION:
MUCH HAS BEEN WRITTEN DESCRIBING STRUCTURED PROGRAMMING (SP) TECHNOLOGIES, BUT LITTLE HAS BEEN WRITTEN CONCERNING MEASURABLE RESULTS FROM USING THOSE TECHNOLOGIES TO DELIVER PROGRAMS WITHIN COST, TIME, AND PERFORMANCE CONSTRAINTS. SP TECHNOLOGIES HAVE BEEN APPLIED TO LARGE-SCALE REAL-TIME PROGRAM DEVELOPMENT, ANALYZED, AND FURTHER DEVELOPED AT HUGHES AIRCRAFT COMPANY SINCE 1971. AS NEW SOFTWARE PROJECTS ARE STARTED, THE RESPECTIVE MANAGERS MUST STATE WHICH OF THE TECHNOLOGIES WILL AND WILL NOT BE USED. THE USE OF THE APPLIED TECHNIQUES IS THEN MONITORED TO DETERMINE BENEFITS AND PROBLEMS AND TO DEFINE USEFUL MODIFICATIONS TO THE TECHNOLOGIES. (A)
5P, 3R.

207 SOFTWARE DESIGN
KATZAN, H., JR. SYSTEMS DESIGN AND DOCUMENTATION -- AN INTRODUCTION TO THE HYPO METHOD. NEW YORK: VAN NOSTRAND REINHOLD, 1976.

208 EMBEDDED TRAINING IN SYSTEM FOR NAIVE USERS
KENNEDY, T.C.S. SOME BEHAVIOURAL FACTORS AFFECTING THE TRAINING OF NAIVE USERS OF AN INTERACTIVE COMPUTER SYSTEM. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1975, 7, 817-834.
DESCRIPTION:
THIS PAPER DESCRIBES THE DESIGN CONSIDERATIONS UNDERLYING THE DEVELOPMENT OF A SELF-CONTAINED COMPUTER SYSTEM WHICH IS TO FORM THE BASIS OF A MEDICAL INFORMATION SYSTEM AT SOUTHEND HOSPITAL. A DETAILED TRIAL HAS BEEN CONDUCTED TO EXAMINE THE PROBLEMS IN TRAINING NAIVE COMPUTER USERS IN THE USE OF SUCH A SYSTEM. THE TRIAL INVOLVED A LARGE SAMPLE OF CLERICAL AND SECRETARIAL STAFF AND PROVIDED 50 HOURS OF OBSERVATION AND MEASUREMENT OF MAN-MACHINE INTERACTION. ANALYSIS OF TEST RESULTS HAS REQUIRED THE DEVELOPMENT OF NEW MEASURES OF PERFORMANCE FOR RECORDING BEHAVIORAL VARIABLES, CONCEPTUALIZATION OF THE SYSTEM, AND LEVEL OF ABILITY.
IT IS SHOWN THAT IT IS POSSIBLE, WITH A SELF-TEACHING COMPUTER SYSTEM, TO TRAIN "COMPUTER-NAIVE" CLERICAL STAFF TO A HIGH DEGREE OF COMPETENCE IN A VERY SMALL NUMBER OF SHORT TRAINING SESSIONS. BEHAVIORAL PATTERNS ARE EXAMINED WITH REGARD TO THEIR INFLUENCE ON THE DESIGN OF COMMAND STRUCTURES. (A)
18P, 12R.

209 PRODUCTION SYSTEMS

KIBLER, D.F., NEIGHBORS, J.M., & STANDISH, T.A. PROGRAM MANIPULATION VIA AN EFFICIENT PRODUCTION SYSTEM. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 163-174 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 163-174).

210 PROGRAMMING

KNUTH, D.E. AN EMPIRICAL STUDY OF FORTRAN PROGRAMS. SOFTWARE-PRACTICE AND EXPERIENCE, 1971, 1, 105-133 (ALSO IBM RESEARCH REPORT RC-3276, IBM CORP., MARCH 1971; TECHNICAL REPORT CS-186, STANFORD, CALIFORNIA: STANFORD UNIVERSITY, COMPUTER SCIENCE DEPARTMENT, 1971). (NTIS NO. AD 715513)

DESCRIPTION:

A SAMPLE OF PROGRAMS, WRITTEN IN FORTRAN BY A WIDE VARIETY OF PEOPLE IN A WIDE VARIETY OF APPLICATIONS, WAS CHOSEN "AT RANDOM" IN AN ATTEMPT TO DISCOVER QUANTITATIVELY "WHAT PROGRAMMERS REALLY DO." STATISTICAL RESULTS OF THIS SURVEY ARE PRESENTED HERE, TOGETHER WITH SOME OF THEIR APPARENT IMPLICATIONS FOR FUTURE WORK IN COMPILER DESIGN. THE PRINCIPAL CONCLUSION WHICH MAY BE DRAWN IS THE IMPORTANCE OF A PROGRAM "PROFILE," NAMELY, A TABLE OF FREQUENCY COUNTS WHICH RECORD HOW OFTEN EACH STATEMENT IS PERFORMED IN A TYPICAL RUN; THERE ARE STRONG INDICATIONS THAT PROFILE-KEEPING SHOULD BECOME A STANDARD PRACTICE IN ALL COMPUTER SYSTEMS, FOR CASUAL USERS, AS WELL AS SYSTEM PROGRAMMERS. THIS PAPER IS THE REPORT OF A THREE-MONTH STUDY UNDERTAKEN BY THE AUTHOR AND ABOUT A DOZEN STUDENTS AND REPRESENTATIVES OF THE SOFTWARE INDUSTRY DURING THE SUMMER OF 1970. IT IS HOPED THAT A READER WHO STUDIES THIS REPORT WILL OBTAIN A FAIRLY CLEAR CONCEPTION OF HOW FORTRAN IS BEING USED, AND WHAT COMPILERS CAN DO ABOUT IT. (A)

42P, 19R.

211 STRUCTURED PROGRAMMING

KNUTH, D.E. A REVIEW OF "STRUCTURED PROGRAMMING" (TECHNICAL REPORT NO. STAN-CS-73-371). STANFORD, CALIFORNIA: STANFORD UNIVERSITY, COMPUTER SCIENCE DEPARTMENT, 1973.

212 STRUCTURED PROGRAMMING

KNUTH, D.E. STRUCTURED PROGRAMMING WITH GO TO STATEMENTS. COMPUTING SURVEYS, 1974, 6, 261-301.

DESCRIPTION:

A CONSIDERATION OF SEVERAL DIFFERENT EXAMPLES SHEDS NEW LIGHT ON THE PROBLEM OF CREATING RELIABLE, WELL-STRUCTURED PROGRAMS THAT BEHAVE EFFICIENTLY. THIS STUDY FOCUSES LARGELY ON TWO ISSUES: (A) IMPROVED SYNTAX FOR ITERATIONS AND ERROR EXITS, MAKING IT POSSIBLE TO WRITE A LARGER CLASS OF PROGRAMS CLEARLY AND EFFICIENTLY WITHOUT GO TO STATEMENTS; (B) A METHODOLOGY OF PROGRAM DESIGN, BEGINNING WITH READABLE AND CORRECT, BUT POSSIBLY INEFFICIENT PROGRAMS THAT ARE SYSTEMATICALLY TRANSFORMED IF NECESSARY INTO EFFICIENT AND CORRECT, BUT POSSIBLY LESS READABLE CODE. THE DISCUSSION BRINGS OUT OPPOSING POINTS OF VIEW ABOUT WHETHER OR NOT GO TO STATEMENTS SHOULD BE ABOLISHED; SOME MERIT IS FOUND ON BOTH SIDES OF THIS QUESTION. FINALLY, AN ATTEMPT IS MADE TO DEFINE THE TRUE NATURE OF STRUCTURED PROGRAMMING, AND TO RECOMMEND FRUITFUL DIRECTIONS FOR FURTHER STUDY. (A)

41P, 102R.

213 PROGRAMMING

KNUTH, D.E. COMPUTER PROGRAMMING AS AN ART. COMMUNICATIONS OF THE ACM, 1974, 17, 667-673.

DESCRIPTION:

MUCH HAS BEEN WRITTEN ABOUT THE NEED FOR COMPUTER PROGRAMMING TO MAKE A TRANSITION FROM AN ART TO A DISCIPLINED SCIENCE. IMPLICIT IN SUCH REMARKS IS THE IDEA THAT AN ACTIVITY THAT IS CLASSIFIED AS AN ART IS LESS DESIRABLE THAN ONE CLASSIFIED AS A SCIENCE. THIS PAPER ATTEMPTS TO DEMONSTRATE THAT COMPUTER PROGRAMMING IS AN ART BECAUSE IT INVOLVES ACCUMMULATED REAL WORLD KNOWLEDGE, REQUIRES SKILL AND INGENUITY, AND PRODUCES OBJECTS OF BEAUTY. A PROGRAMMER WHO VIEWS HIMSELF AS AN ARTIST WILL ENJOY WHAT HE DOES AND WILL DO IT BETTER. (AEA)

7P, 35R.

214 PROGRAMMING LANGUAGES

KNUTH, D.E., & PARDO, L.T. THE EARLY DEVELOPMENT OF PROGRAMMING LANGUAGES (TECHNICAL REPORT STAN-CS-76-562). STANFORD, CALIFORNIA: STANFORD UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, AUGUST 1976. (NTIS NO. AD AC32123)

DESCRIPTION:

THIS PAPER SURVEYS THE EVOLUTION OF HIGH LEVEL PROGRAMMING LANGUAGES DURING THE FIRST DECADE OF COMPUTER PROGRAMMING ACTIVITY. WE DISCUSS THE CONTRIBUTIONS OF ZUSE (PLANKALKUEL, 1945), GOLDSTINE/VON NEUMANN (FLOW DIAGRAMS, 1946), CURRY (COMPOSITION, 1946), MAUCHLY ET AL (SHORT CODE, 1949), BURKS, (INTERMEDIATE PL, 1950), RUTISHAUSER (1951), BOEHM (1951), GLENNIE (AUTOCODE, 1952), HOPPER ET AL (A-2, 1953), LANING/ZIERLER (1953), BACKUS ET AL (FORTRAN, 1954-1957), BROOKER (MARK I AUTOCODE, 1954), KAMQUIN/ LIUBIMSKII (PI-PI-2, 1954), ERSHOV (PI-PI, 1955), GREMS/PORTER (BACAIC, 1955), ELSWORTH ET AL (KOMPILER 2, 1955), BLUM (ADES, 1956), PORLIS ET AL (IT, 1955), KATZ ET AL (MATH-MATIC, 1956-1958), HOPPER ET AL.(FLOW-MATIC, 1956-1958), BAUER/SAMELSON (1956-1958). THE PRINCIPAL FEATURES OF EACH CONTRIBUTION ARE ILLUSTRATED; AND FOR PURPOSES OF COMPARISON, A PARTICULAR FIXED ALGORITHM HAS BEEN ENCODED (AS FAR AS POSSIBLE) IN EACH OF THE LANGUAGES. THIS RESEARCH IS BASED PRIMARILY ON UNPUBLISHED SOURCE MATERIALS, AND THE AUTHORS HOPE THAT THEY HAVE BEEN ABLE TO COMPILE A FAIRLY COMPLETE PICTURE OF THE EARLY DEVELOPMENTS IN THIS AREA. (A)

11P, 127R.

215 SOFTWARE DEVELOPMENT

KRALY, T.P., NAUGHTON, J.J., SMITH, R.L., & TINAKOFF, N. STRUCTURED PROGRAMMING SERIES (VOL. 8): PROGRAM DESIGN STUDY; FINAL REPORT (REPORT NO. RADC-TR-74-300-VOL-8). GRIFFISS AFB, NEW YORK: ROME AIR DEVELOPMENT CENTER, MAY 1975. (NTIS NO. AD A016415)

DESCRIPTION:

THIS VOLUME REPORTS ON PROGRAM DESIGN TOOLS AND TECHNIQUES. IT EVALUATES THESE TOOLS AND TECHNIQUES DETERMINING THEIR RELATIONSHIP TO STRUCTURED PROGRAMMING TECHNOLOGY AND PRESENTS THEIR ADVANTAGES AND DISADVANTAGES. THE MAJOR CONCLUSION OF THIS REPORT IS THAT PROGRAM DESIGN LANGUAGES, AS WELL AS DETAILED FLOWCHARTS, FACILITATE EXPRESSION OF PROGRAM CONTROL FLOW BUT THE FORMER IS MORE EASILY PRODUCED AND MAINTAINED AT LESS EXPENSE. (A)

216 PROGRAMMING

KREITZBERG, C.B., & SHNEIDEPMAN, B. THE ELEMENTS OF FORTRAN STYLE: TECHNIQUES FOR EFFECTIVE PROGRAMMING. NEW YORK, NEW YORK: HARCOURT BRACE, 1972.

DESCRIPTION:

THE ATTRIBUTES OF A WELL-WRITTEN COMPUTER PROGRAM ARE EVIDENT. IT IS EFFICIENT, FAST, WELL-DOCUMENTED, ELEGANT, AND, OF COURSE, CORRECT. HOW TO WRITE SUCH A PROGRAM IS NOT SO OBVIOUS. BY PRESENTING ESSENTIAL TECHNIQUES OF FORTRAN STYLE, THIS BOOK TEACHES THE ART OF WRITING A GOOD PROGRAM. EACH TECHNIQUE IS CAREFULLY EXPLAINED AND THE RATIONALE FOR IT IS GIVEN SO THAT THE STUDENT CAN LEARN WHEN TO APPLY A RULE AND WHEN TO DEVELOP HIS OWN. IT IS OUR HOPE THAT "THE ELEMENTS OF FORTRAN STYLE" WILL ENABLE THE NOVICE PROGRAMMER TO BECOME A GOOD ONE AND THE GOOD PROGRAMMER TO BECOME A BETTER ONE. (A, ABBR.)

127P, 23R.

217 PROGRAMMING STYLE

KREITZBERG, C.B., & SHNEIDERMAN, B. THE ELEMENTS OF PROGRAMMING STYLE. MODERN DATA, AUGUST 1972, 5, 40-41.

218 PROGRAMMING, INSTRUCTION

KREITZBERG, C., & SWANSON, L. A COGNITIVE MODEL FOR STRUCTURING AN INTRODUCTORY PROGRAMMING CURRICULUM. AFIPS CONFERENCE PROCEEDINGS, 1974, 43, 307-311.

DESCRIPTION:

INCREASING ATTENTION IS BEING GIVEN TO UNDERGRADUATE TRAINING IN COMPUTER USE AND "COMPUTER LITERACY" IS RAPIDLY BECOMING A REQUIREMENT OF THE EDUCATED PERSON. WHILE THE DEMAND FOR PROGRAMMING INSTRUCTION IS INCREASING, HOWEVER, LITTLE SYSTEMATIC ATTENTION HAS BEEN GIVEN TO HOW THAT INSTRUCTION MIGHT BE IMPROVED. IN THIS PAPER, WE WILL DISCUSS THREE FACTORS THAT COULD LEAD TO MORE MEANINGFUL INSTRUCTION -- PROVIDING SUFFICIENT CONTEXT, FACILITATING VERTICAL TRANSFER OF CONCEPTS AND REDUCING "COMPUTER SHOCK." EMPIRICAL EVIDENCE IS REQUIRED TO SUPPORT THESE HYPOTHESES. (MEA)

5P, 13R.

219 ALTERNATIVE DATA BASE MODELS

KUHN, M., & SHNEIDERMAN, B. TWO EXPERIMENTAL COMPARISONS OF RELATIONAL AND HIERARCHICAL DATABASE MODELS (IFSM TECHNICAL REPORT NO. 31). COLLEGE PARK, MARYLAND: UNIVERSITY OF MARYLAND, DEPARTMENT OF INFORMATION SYSTEMS MANAGEMENT, FEBRUARY 1978.

220 SOFTWARE PHYSICS

KULM, G. LANGUAGE LEVEL APPLIED TO THE INFORMATION CONTENT OF TECHNICAL PROSE. IN N.A. CHIGIER, & E.A. STERN (EDS.) COLLECTIVE PHENOMENA AND THE APPLICATION OF PHYSICS TO OTHER FIELDS OF SCIENCE. FAYETTEVILLE, NEW YORK: BRAIN RESEARCH PUBLICATIONS, 1975.

DESCRIPTION:

ENGLISH PASSAGES WERE USED TO ILLUSTRATE THE ABILITY OF A FORMAL MEASURE OF LANGUAGE LEVEL TO DISCRIMINATE PASSAGES WRITTEN AT DIFFERENT LEVELS OF DIFFICULTY. THE RESULTS OF THE EXPERIMENTS SHOWED THAT THE MEASURES OF LANGUAGE LEVEL AND INFORMATION CONTENT APPEAR TO HAVE PROMISE IN THEIR APPLICATION TO TECHNICAL ENGLISH. (O)

221 TIME-SHARING VS. BATCH PROCESSING

LAMPSON, B.W. A CRITIQUE OF "AN EXPLORATORY INVESTIGATION OF PROGRAMMER PERFORMANCE UNDER ON-LINE AND OFF-LINE CONDITIONS". IEEE TRANSACTIONS ON HUMAN FACTORS IN ELECTRONICS, 1967, HFE-8, 48-51.

DESCRIPTION:

THE PAPER BY GRANT AND SACKMAN (1967), "AN EXPLORATORY INVESTIGATION OF PROGRAMMER PERFORMANCE UNDER ON-LINE AND OFF-LINE CONDITIONS" IS DISCUSSED CRITICALLY. PRIMARY EMPHASIS IS ON THIS PAPER'S FAILURE TO CONSIDER THE MEANING OF THE NUMBERS OBTAINED. AN UNDERSTANDING OF THE NATURE OF AN ON-LINE SYSTEM IS NECESSARY FOR PROPER INTERPRETATION OF THE OBSERVED RESULTS FOR DEBUGGING TIME, AND THE RESULTS FOR COMPUTER TIME ARE CRITICALLY DEPENDENT ON THE IDIOSYNCRACIES OF THE SYSTEM ON WHICH THE WORK WAS DONE. LACK OF ATTENTION TO THESE MATTERS CANNOT BE COMPENSATED FOR BY ANY AMOUNT OF STATISTICAL ANALYSIS. FURTHERMORE, MANY OF THE CONCLUSIONS DRAWN AND SUGGESTIONS MADE ARE TOO VAGUE TO BE USEFUL. (A)

4P, 11R.

222 SOFTWARE DEVELOPMENT

LAPADULA, L.J. ENGINEERING OF QUALITY SOFTWARE SYSTEMS (SOFTWARE RELIABILITY MODELING AND MEASUREMENT TECHNIQUES) (REPORT NUMBER MTR-2648-VOL-8). BEDFORD, MASSACHUSETTS: MITRE CORP., JANUARY 1975

223 PROGRAMMING LANGUAGES

LEDGARD, H.F. TEN MINI-LANGUAGES: A STUDY OF TOPICAL ISSUES IN PROGRAMMING LANGUAGES. COMPUTING SURVEYS, 1971, 3, 115-146.

DESCRIPTION:

THE PROLIFERATION OF PROGRAMMING LANGUAGES HAS RAISED MANY ISSUES OF LANGUAGE DESIGN, DEFINITION, AND IMPLEMENTATION. THIS PAPER PRESENTS A SERIES OF TEN MINI-LANGUAGES, EACH OF WHICH EXPOSES SALIENT FEATURES FOUND IN EXISTING PROGRAMMING LANGUAGES. THE VALUE OF THE MINI-LANGUAGES LIES IN THEIR BREVITY OF DESCRIPTION AND THE ISOLATION OF IMPORTANT LINGUISTIC FEATURES: IN PARTICULAR, THE NOTIONS OF ASSIGNMENT, TRANSFER OF CONTROL, FUNCTIONS, PARAMETER PASSING, TYPE CHECKING, DATA STRUCTURES, STRING MANIPULATION, AND INPUT/OUTPUT. THE MINI-LANGUAGES MAY SERVE A VARIETY OF USES: NOTABLY, AS A PEDAGOGICAL TOOL FOR TEACHING PROGRAMMING LANGUAGES, AS A SUBJECT OF STUDY FOR THE DESIGN OF PROGRAMMING LANGUAGES, AND AS A SET OF TEST CASES FOR METHODS OF LANGUAGE IMPLEMENTAION OR FORMAL DEFINITION. (A)

32P, 16R.

224 STRUCTURED PROGRAMMING

LEDGARD, H.F. THE CASE FOR STRUCTURED PROGRAMMING. BIT, 1973, 13, 45-57.

DESCRIPTION:

THIS REPORT IS MAINLY A RESPONSE TO A PAPER BY HENDERSON AND SNOWDEN, "AN EXPERIMENT IN STRUCTURED PROGRAMMING." THE NOTIONS OF STRUCTURED PROGRAMMING, TOP-DOWN PROGRAMMING, AND STEPWISE REFINEMENT ARE COMPARED, AND SOME CAREFUL GUIDELINES FOR THE PROPER USE OF STRUCTURED PROGRAMMING APPROACHES ARE SUGGESTED. (A)

13P, 6R.

225 PROGRAM DESIGN

LEVIN, S.L. A SHORT SURVEY OF MODELS IN THE DESIGN PROCESS (TECHNICAL REPORT NO. 71). IRVINE, CALIFORNIA: UNIVERSITY OF CALIFORNIA, DEPARTMENT OF INFORMATION AND COMPUTER SCIENCE, 1975.

DESCRIPTION:

A TAXONOMY FOR DESCRIBING MODELS OF THE DESIGN PROCESS IS PRESENTED. THE TAXONOMY IS USED IN COMPARING AND DISCUSSING SEVERAL TYPES OF DESIGN MODELS. THE PAPER'S APPENDIX CONTAINS A SUMMARY DESCRIPTION FOR EACH MODEL THAT IS DISCUSSED. (A)

226 SOFTWARE DESIGN

LEVIN, S.L. PROBLEM SELECTION IN SOFTWARE DESIGN (TECHNICAL REPORT NO. 93). IRVINE, CALIFORNIA: UNIVERSITY OF CALIFORNIA, DEPARTMENT OF INFORMATION AND COMPUTER SCIENCE, NOVEMBER 1976.

DESCRIPTION:

THIS PAPER REPORTS THE RESULTS OF RESEARCH INTO THE COGNITIVE PROCESSES OF SOFTWARE DESIGN. DESIGN IS VIEWED AS A COMPLEX ACTIVITY INVOLVING THREE FUNDAMENTAL PROCESSES: SELECTING PROBLEMS TO WORK ON, GATHERING NEEDED INFORMATION FOR THEIR SOLUTION, AND GENERATING SOLUTIONS. A DETAILED MODEL OF THE PROBLEM SELECTION PROCESS IS PRESENTED. (A, ABBR.)
96P, 19R.

227 ERRORS, PROGRAMMING

LIPOW, M. ESTIMATION OF SOFTWARE PACKAGE RESIDUAL ERROR (TECHNICAL REPORT NO. TRW-SS-72-09). REDONDO BEACH, CALIFORNIA: TRW, NOVEMBER 1972.

DESCRIPTION:

A METHOD FOR ESTIMATING THE NUMBER OF ERRORS REMAINING IN A SOFTWARE PACKAGE IS PROPOSED AND ANALYZED. IT IS BASED UPON A SCHEME, PROPOSED BY H.D. MILLS OF IBM, IN WHICH A SET OF KNOWN ERRORS IS 'SEEDED' INTO THE SOFTWARE. A SPECIFIED NUMBER OF TESTS IS CONDUCTED, EACH TEST CAPABLE OF FINDING ONE OF THE INDIGENOUS OR UNKNOWN ERRORS, OR ONE OF THE SEEDED ERRORS WITH THE SAME PROBABILITY, OR OF FINDING NO ERROR. THE PROBLEM IS TO ESTIMATE THE TWO PARAMETERS: N1, THE UNKNOWN NUMBER OF RESIDUAL ERRORS, AND Q, THE UNKNOWN PROBABILITY OF DETECTING EITHER A SEEDED OR AN INDIGENOUS ERROR. THE REASON THE SEEDING SCHEME WORKS, OF COURSE, IS BECAUSE OBSERVED DETECTION RATE OF SEEDED ERRORS WITH A KNOWN NUMBER OF SEEDED ERRORS REMAINING YIELDS INFORMATION ON THE NUMBER OF RESIDUAL INDIGENOUS ERRORS, SINCE THE LATTER ERRORS ARE DETECTED AT THE SAME RATE, BY ASSUMPTION. IN ADDITION TO FINDING MAXIMUM LIKELIHOOD ESTIMATORS FOR N1 AND Q, THE CONDITIONAL DISTRIBUTION OF THE MAXIMUM LIKELIHOOD ESTIMATOR FOR N1, GIVEN THE TOTAL NUMBER OF ERRORS DETECTED, IS GIVEN ANALYTICALLY AND ALSO COMPUTED IN TWO CASES. THE DISTRIBUTION CALCULATION ALSO YIELDS THE MEAN AND THE ROOT MEAN SQUARE DEVIATION OF THE ESTIMATOR FOR N1, AND CONSEQUENTLY, AN INDICATION OF ITS BIAS AND PRECISION FOR VARIOUS TRUE VALUES OF N1, NUMBER OF SEEDED ERRORS, AND OBSERVED TOTAL NUMBERS OF ERRORS. (A)
13P, 2R.

228 SOFTWARE METRICS

LIPOW, M. MEASUREMENT OF COMPUTER SOFTWARE -- FINDINGS AND RECOMMENDATIONS OF THE JOINT LOGISTICS COMMANDERS SOFTWARE RELIABILITY WORK GROUP (VOL. 1). NOVEMBER 1975.

229 PROGRAMMING, ERRORS

LITECKY, C.R. A STUDY OF ERRORS, ERROR-PRONENESS, AND ERROR DIAGNOSIS OF PROGRAMMING LANGUAGES WITH SPECIAL REFERENCE TO COBOL. DOCTORAL DISSERTATION, UNIVERSITY OF MINNESOTA, MINNEAPOLIS, MINNESOTA, 1974. (UNIVERSITY MICROFILMS NO. 74-17, 263)

DESCRIPTION:

AN EXPERIMENT WAS CONDUCTED TO EXAMINE THE TYPES AND FREQUENCIES OF ERRORS MADE BY BEGINNING COBOL PROGRAMMERS. SUGGESTIONS ARE MADE FOR IMPROVEMENTS IN THE COBOL LANGUAGE, COBOL COMPILERS, AND AUTOMATIC ERROR CORRECTION. (MEA)

199P, 47R.

230 PROGRAMMING

LITECKY, C.R. ASSISTING STUDENTS TO AVOID CODING ERRORS IN THE COBOL LANGUAGE. AEDS JOURNAL, 1976 (WINTER), 36-38.

DESCRIPTION:

COBOL IS THE MOST WIDELY USED PROGRAMMING LANGUAGE FOR DATA PROCESSING; MOST COMMERCIAL CODING IS DONE IN COBOL. YET COBOL HAS NOT BEEN TAUGHT AS EXTENSIVELY AS MIGHT BE EXPECTED. PERHAPS THE MAJOR REASON IS THAT THE COBOL LANGUAGE SEEMS TO HAVE A PARTICULARLY TROUBLESOME AND ERROR-PRONE SYNTAX FOR INEXPERIENCED PROGRAMMING LANGUAGE LEARNERS.

THIS SHORT PAPER PRESENTS DATA ON COBOL SYNTACTICAL ERROR FREQUENCIES AND AN APPROACH THAT INSTRUCTORS MAY USE TO ASSIST STUDENTS TO AVOID ERRORS WHILE LEARNING THE COBOL LANGUAGE. THESE DATA MAY ALSO BE HELPFUL IN FINDING ERRORS DURING DEBUGGING OF SYNTACTICAL ERRORS IN COBOL PROGRAMS.

231 PROGRAMMING LANGUAGES

LITECKY, C.R., & DAVIS, G.B. A STUDY OF ERRORS, ERROR-PRONENESS, AND ERROR DIAGNOSIS IN COBOL. COMMUNICATIONS OF THE ACM, 1976, 19, 33-37.

DESCRIPTION:

THIS PAPER PROVIDES DATA ON COBOL ERROR FREQUENCY FOR CORRECTION OF ERRORS IN STUDENT-ORIENTED COMPILERS, IMPROVEMENT OF TEACHING, AND CHANGES IN PROGRAMMING LANGUAGE. COBOL WAS STUDIED BECAUSE OF ECONOMIC IMPORTANCE, WIDESPREAD USAGE, POSSIBLE ERROR-INDUCING DESIGN, AND LACK OF RESEARCH. THE TYPES OF ERRORS WERE IDENTIFIED IN A PILOT STUDY; THEN, USING THE 132 ERROR TYPES FOUND, 1,777 ERRORS WERE CLASSIFIED IN 1,400 RUNS OF 73 COBOL STUDENTS. ERROR DENSITY WAS HIGH: 20 PERCENT OF THE TYPES CONTAINED 80 PERCENT OF THE TOTAL FREQUENCY, WHICH IMPLIES HIGH POTENTIAL EFFECTIVENESS FOR SOFTWARE-BASED CORRECTION OF COBOL. SURPRISINGLY, ONLY FOUR HIGH-FREQUENCY ERRORS WERE ERROR-PRONE, WHICH IMPLIES MINIMAL ERROR INDUCING DESIGN. 80 PERCENT OF COBOL MISSPELLINGS WERE CLASSIFIABLE IN THE FOUR ERROR CATEGORIES OF PREVIOUS RESEARCHERS, WHICH IMPLIES THAT COBOL MISSPELLINGS ARE CORRECTABLE BY EXISTENT ALGORITHMS. RESERVED WORD USAGE WAS NOT ERROR-PRONE, WHICH IMPLIES MINIMAL INTERFERENCE WITH USAGE OF RESERVED WORDS. OVER 80 PERCENT OF ERROR DIAGNOSIS WAS FOUND TO BE INACCURATE. SUCH FEEDBACK IS NOT OPTIMAL FOR USERS, PARTICULARLY FOR THE LEARNING USER OF COBOL. (A)

9P, 16R.

232 ATTITUDES TOWARD SOFTWARE MAINTENANCE

LIU, C.C. A LOOK AT SOFTWARE MAINTENANCE. DATAMATION, NOVEMBER 1976, 22(11), 51-55.

233 DEBUGGING

LOESER, R., & GAPOSCHKIN, E.M. THE SECOND LAW OF DEBUGGING. SOFTWARE-PRACTICE AND EXPERIENCE, 1976, 6, 577-578.

DESCRIPTION:

DEBUGGING EFFICIENCY DEPENDS ON SEARCHING FOR THE BUG IN THE RIGHT PLACE. ON MANY OCCASIONS, WHEN CONFRONTED BY PARTICULARLY STUBBORN BUGS, THE PRINCIPLE: 'IF YOU DON'T SEE THE BUG WHERE YOU'RE LOOKING, THEN YOU'RE LOOKING IN THE WRONG PLACE,' HAS HELPED US GREATLY. WE HAVE COME TO REFER TO THIS PRINCIPLE AS THE SECOND LAW OF DEBUGGING. OUR PRACTICAL EXPERIENCE WITH IT EXTENDS OVER A DOZEN YEARS. ON MANY OCCASIONS (JUDGING AFTER THE FACT), USE OF THIS LAW ENABLED US TO LOCATE A BUG MORE QUICKLY THAN WE OTHERWISE WOULD HAVE DONE. ON MANY OTHER OCCASIONS (AGAIN JUDGING AFTER THE FACT), WE WOULD HAVE FOUND A PARTICULAR BUG MORE QUICKLY IF WE HAD USED IT. WE ARE THOROUGHLY CONVINCED THAT THE SECOND LAW IS A KEY TO MORE RAPID DEBUGGING. IT, THEREFORE, MERITS WIDER RECOGNITION AND APPLICATION. (A)

234 PROGRAMMING

LOVE, L.T. RELATING INDIVIDUAL DIFFERENCES IN COMPUTER PROGRAMMING PERFORMANCE TO HUMAN INFORMATION PROCESSING ABILITIES. UNPUBLISHED DOCTORAL DISSERTATION, UNIVERSITY OF WASHINGTON, 1976.

235 SOFTWARE DEVELOPMENT PROCESS

LOVE, L.T. A REVIEW OF THE VARIABLES WHICH INFLUENCE THE SOFTWARE DEVELOPMENT PROCESS (TECHNICAL REPORT NO. 76ISP001). ARLINGTON, VIRGINIA: GENERAL ELECTRIC COMPANY, 1976.

DESCRIPTION:

THE SOFTWARE DEVELOPMENT PROCESS IS DIVIDED INTO THREE STAGES -- TASK DEMANDS, PROGRAM DEVELOPMENT PROCESS, AND OPERATING PROGRAM. THE PROGRAM DEVELOPMENT PROCESS IS FURTHER DIVIDED INTO FOUR COMPONENTS -- PROGRAMMER, COMPUTER SYSTEM, WORK ENVIRONMENT, AND SOFTWARE DEVELOPMENT SYSTEM.

WITHIN THIS FRAMEWORK, RECENT WORK IN SOFTWARE ENGINEERING AND PSYCHOLOGY ARE REVIEWED AND DISCUSSED. MOST OF THE GENERAL PAPERS IN THE FIELD OF SOFTWARE ENGINEERING ARE INCLUDED IN THIS REVIEW. (A)

45P, 69R.

236 SOFTWARE PHYSICS

LOVE, L.T., & BOWMAN, A.B. AN INDEPENDENT TEST OF THE THEORY OF SOFTWARE PHYSICS. SIGPLAN NOTICES, NOVEMBER 1976, 11(11), 42-49.

DESCRIPTION:

RECENT WORK IN THE FIELD OF SOFTWARE PHYSICS HAS PRODUCED SEVERAL HYPOTHESES RELATING THE NATURE OF ALGORITHMS TO MEASURABLE PROPERTIES OF COMPUTER PROGRAMS. ONE HYPOTHESIS IS THAT HALSTEAD'S MEASURE OF E, THE NUMBER OF ELEMENTARY MENTAL DISCRIMINATIONS REQUIRED TO IMPLEMENT AN ALGORITHM, IS STRONGLY RELATED TO MEASURABLE PROPERTIES OF COMPUTER PROGRAMS. SEVERAL EXPERIMENTS HAVE SHOWN A SURPRISINGLY HIGH CORRELATION BETWEEN E AND SUCH MEASURABLE PROPERTIES OF PROGRAMS AS NUMBER OF BUG , CODING TIMES, ETC. THIS PAPER PRESENTS THE RESULTS OF AN INDEPENDENT STUDY TO TEST THIS HYPOTHESIS.

HALSTEAD'S MEASURE IS CALCULATED FOR THE TEST PROGRAMS USED IN TWO STUDIES PUBLISHED BY OTHER INVESTIGATORS (GOULD, 1975; WEISSMAN, 1974). CORRELATIONS ARE GIVEN BETWEEN E AND GOULD'S MEDIAN DEBUG TIME (R=.20) AND AVERAGE NUMBER OF ERRORS (R=.78), AND BETWEEN E AND WEISSMAN'S COMPREHENSION QUIZZES AND SELF-EVALUATIONS (CORRELATIONS RANGE FROM -.97 TO +.37). (A & HRR)

3P, 11R.

237 SOFTWARE DEVELOPMENT PROCESS

LOVE, L.T., & FITZSIMMONS, A. A SURVEY OF SOFTWARE PRACTITIONERS TO IDENTIFY CRITICAL FACTORS IN THE SOFTWARE DEVELOPMENT PROCESS (TECHNICAL REPORT NO. 76D6AS1). ARLINGTON, VIRGINIA: GENERAL ELECTRIC COMPANY, INFORMATION SYSTEMS PROGRAMS, 1976.

DESCRIPTION:

WE MUST FIRST IDENTIFY THE CRITICAL AND INEFFICIENT COMPONENTS OF THE SOFTWARE DEVELOPMENT PROCESS, THEN DETERMINE WHICH ONES CAN BE ELIMINATED OR IMPROVED. THE PRESENT RESEARCH PROJECT WAS CONCEIVED AS A STEP TOWARD THE IDENTIFICATION OF CRITICAL COMPONENTS OF THE SOFTWARE DEVELOPMENT PROCESS.

ESSENTIALLY, WE DECIDED TO POLL THOSE PEOPLE MOST QUALIFIED TO PINPOINT CURRENT SOFTWARE DEVELOPMENT DIFFICULTIES; VIZ., THE SOFTWARE DEVELOPERS THEMSELVES. WE RECOGNIZE THAT THE INTROSPECTIONS OF SOFTWARE PRACTITIONERS MAY NOT GENERATE THE ULTIMATE ANSWERS SINCE THE DIFFICULTIES WHICH THEY BELIEVE TO BE MOST CRITICAL MAY NOT, IN THE FINAL ANALYSIS, BE THE MOST CRITICAL. WE, THEREFORE, MAKE NO PRETENSE THAT WE ARE FINDING THE ANSWER TO OUR QUESTION. RATHER, UPON CONCLUSION OF THIS EFFORT, WE CAN ONLY SAY THAT WE KNOW WHAT SOFTWARE PRACTITIONERS 'THINK' THE ANSWER IS.

A SIGNIFICANT CONTRIBUTION OF THIS WORK WILL BE THE ABILITY TO NOT ONLY IDENTIFY THE CRITICAL COMPONENTS OF THE SOFTWARE DEVELOPMENT PROCESS, BUT ALSO TO HAVE AN INDEX OF THE CRITICALITY OF EACH SUCH COMPONENT.

POPULATION SAMPLED: TWO HUNDRED SURVEYS WERE SENT OUT TO SOFTWARE DEVELOPERS AND MANAGERS IN FIVE LOCATIONS OF GENERAL ELECTRIC. THESE INCLUDED 50 SENT TO ARLINGTON; 60 SENT TO SUNNYVALE; 30 TO BELTSVILLE; 10 TO SCHENECTADY; AND 50 TO SYRACUSE. THE RESPONDENTS INCLUDED ANY PERSON WHO WAS OR HAD BEEN DIRECTLY INVOLVED IN DEVELOPING SOFTWARE FOR AT LEAST ONE YEAR. THE RESPONDENTS WERE INSTRUCTED TO ANSWER EACH QUESTION BASED ON THEIR OWN EXPERIENCES IN SOFTWARE DEVELOPMENT PROJECTS. BECAUSE OF THE RAPID RATE OF CHANGE OF METHODS AND TOOLS IN THE SOFTWARE FIELD, THE RESPONDENTS WERE ASKED THAT RESPONSES BE LIMITED TO EXPERIENCES DURING THE LAST TWO YEARS.

DESCRIPTION OF SURVEY: THE SURVEY WAS COMPOSED OF THREE SECTIONS AND A COVER LETTER DESCRIBING THE PURPOSE AND GENERAL OUTLINE OF THE SURVEY. (A COPY OF THE COMPLETE SURVEY IS IN APPENDIX A.) (A)

238 DEBUGGING

LOVE, R.E. OPTIMISING COMPUTER SOURCE LANGUAGE USING A VISUAL DISPLAY. IN PROCEEDINGS OF THE INTERNATIONAL SYMPOSIUM ON MAN-MACHINE SYSTEMS, 8-12 SEPTEMBER 1969 (VOL. 1). IEEE CONFERENCE RECORD NO. 69C58-MMS, INSTITUTE OF ELECTRICAL AND ELECTRONIC ENGINEERS.

239 SOFTWARE DEVELOPMENT PROCESS

LOVE, T., & FITZSIMMONS, A. A SURVEY OF SOFTWARE PRACTITIONERS TO IDENTIFY CRITICAL FACTORS IN THE SOFTWARE DEVELOPMENT PROCESS (REPORT NO. 77ISP002). ARLINGTON, VIRGINIA: GENERAL ELECTRIC COMPANY, INFORMATION SYSTEMS PROGRAMS, JANUARY, 1977.

DESCRIPTION:

IT HAS BECOME INCREASINGLY IMPORTANT WITHIN GE TO IMPROVE THE PRODUCTIVITY OF PROGRAMMERS AS WELL AS THE RELIABILITY OF THE SOFTWARE THEY PRODUCE. TO PROVIDE US WITH INFORMATION AS TO WHICH COMPONENTS OF THE SOFTWARE DEVELOPMENT PROCESS ARE MOST IMPORTANT TO THE OVERALL SUCCESS OF A SOFTWARE DEVELOPMENT EFFORT, WE SURVEYED 89 SOFTWARE PRACTITIONERS FROM 4 GE LOCATIONS.

WE ATTEMPTED TO ELICIT THE SAME TYPE OF INFORMATION IN THREE DIFFERENT WAYS: 1) RATINGS OF THE CRITICALITY OF 100 INDIVIDUAL PROBLEMS, 2) RANK ORDERING OF CLASSES OF PROBLEMS, AND 3) OPEN-ENDED QUESTIONS REQUESTING SIMILAR INFORMATION.

NO SINGLE PROBLEM STOOD OUT AS MOST CRITICAL IN ALL THREE SECTIONS OF THE SURVEY. IN SECTION 1, MOST CRITICAL DIFFICULTIES WERE RELATED TO THE MANAGEMENT PLAN AND REQUIREMENTS ANALYSIS. BY CONTRAST, ON THE OPEN-ENDED QUESTIONS, THE SINGLE MOST IMPORTANT FACTOR CONTRIBUTING TO THE SUCCESS OF SOFTWARE DEVELOPMENT EFFORTS WAS CLAIMED TO BE COMPETENT AND COOPERATIVE PEOPLE. (A)

40P, 3R.

240 SOFTWARE DEVELOPMENT

LUPPINO, F.M., & SMITH, R.L. STRUCTURED PROGRAMMING SERIES (VOL. 5): PROGRAMMING SUPPORT LIBRARY (PSL): FUNCTIONAL REQUIREMENTS: FINAL REPORT (REPORT NO. RADC-TR-74-300-VOL-5). GRIFFIS AFB, NEW YORK: ROME AIR DEVELOPMENT CENTER, JULY 1974. (NTIS NO. AD A003339)

DESCRIPTION:

THIS REPORT DESCRIBES THE FUNCTIONAL REQUIREMENTS FOR A PROGRAMMING SUPPORT LIBRARY. PROGRAMMING SUPPORT LIBRARIES ARE USED TO SUPPORT THE DEVELOPMENT AND MAINTENANCE OF COMPUTER PROGRAMS. THIS REPORT CONTAINS A GENERAL DESCRIPTION OF THE FUNCTIONAL REQUIREMENTS FOR A SPECIFIC PROGRAMMING SUPPORT LIBRARY. EIGHT GENERAL FUNCTIONAL AREAS AND THE REQUIREMENTS RELATED TO ON-LINE IMPLEMENTATION ARE DESCRIBED. THE REPORT ALSO CONTAINS HIPO (HIERARCHY INPUT PROCESS OUTPUT) CHARTS WHICH PROVIDE A GRAPHICAL REPRESENTATION OF THE FUNCTIONAL REQUIREMENTS. (A)

55P, 5R.

241 LEARNING OF PROCEDURES

LYON, D., & THOMAS, J.C., JR. PREDICTING INSUFFICIENT LEARNING OF A COMPLEX PROCEDURE (TECHNICAL REPORT RC-6627). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, JULY 1977. (NTIS NO AD A055586)

242 INFORMATION SYSTEMS

MADNICK, S.E. TRENDS IN COMPUTERS AND COMPUTING: THE INFORMATION UTILITY. SCIENCE, 1977, 195, 1191-1199.

DESCRIPTION:

THE COMPLEXITY, INTERDEPENDENCE, AND RAPIDITY OF EVENTS IN MODERN SOCIETY HAVE ACCELERATED DEMANDS FOR MORE EFFECTIVE WAYS TO STORE, PROCESS, AND MANAGE INFORMATION. ADVANCES IN BOTH COMPUTER HARDWARE (ELECTRONICS) AND SOFTWARE (PROGRAMMING) HAVE PROVIDED THE TECHNOLOGY THAT CAN MAKE IT POSSIBLE TO EFFECTIVELY ADDRESS MANY OF THESE DEMANDS. IN THIS ARTICLE, WE REVIEW THE STRUCTURE OF CLASSICAL COMPUTER-BASED INFORMATION SYSTEMS AND THEN CONSIDER THESE ADVANCES AND SHOW HOW THEY FIT INTO THE EVOLUTION OF THE INFORMATION UTILITY. (A)

243 PROGRAM SYNTHESIS

MANNA, Z., & WALDINGER, R. THE AUTOMATIC SYNTHESIS OF RECURSIVE PROGRAMS. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 29-36 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 29-36).

DESCRIPTION:

WE DESCRIBE A DEDUCTIVE TECHNIQUE FOR THE AUTOMATIC CONSTRUCTION OF RECURSIVE PROGRAMS TO MEET GIVEN INPUT-OUTPUT SPECIFICATIONS. THESE SPECIFICATIONS EXPRESS WHAT CONDITIONS THE OUTPUT OF THE DESIRED PROGRAM IS EXPECTED TO SATISFY. THE DEDUCTIVE TECHNIQUE INVOLVES TRANSFORMING THE SPECIFICATIONS BY A COLLECTION OF RULES, SUMMONED BY PATTERN-DIRECTED FUNCTION INVOCATION. SOME OF THESE TRANSFORMATION RULES EXPRESS THE SEMANTICS OF THE SUBJECT DOMAIN; OTHERS REPRESENT MORE GENERAL PROGRAMMING TECHNIQUES. THE RULES THAT INTRODUCE CONDITIONAL EXPRESSIONS AND RECURSIVE CALLS INTO THE PROGRAM ARE DISCUSSED IN SOME DETAIL.

THE DEDUCTIVE TECHNIQUES DESCRIBED ARE EMBEDDED IN A RUNNING SYSTEM CALLED SYNSYS. THIS SYSTEM ACCEPTS SPECIFICATIONS EXPRESSED IN HIGH-LEVEL DESCRIPTIVE LANGUAGE AND ATTEMPTS TO TRANSFORM THEM INTO A CORRESPONDING LISP PROGRAM. THE TRANSFORMATION RULES ARE EXPRESSED IN THE QLISP PROGRAMMING LANGUAGE. THE SYNTHESIS OF TWO PROGRAMS PERFORMED BY THE SYSTEM ARE PRESENTED. (A)

3P, 13R.

244 PROGRAMMING

MAYER, R.E. DIFFERENT PROBLEM-SOLVING COMPETENCIES ESTABLISHED IN LEARNING COMPUTER PROGRAMMING WITH AND WITHOUT MEANINGFUL MODELS. JOURNAL OF EDUCATIONAL PSYCHOLOGY, 1975, 67, 725-734 (ALSO INSTRUCTIONAL VARIABLES IN MEANINGFUL LEARNING OF COMPUTER PROGRAMMING, INDIANA MATHEMATICAL PSYCHOLOGY PROGRAM REPORT NO. 75-1, BLOOMINGTON, INDIANA: INDIANA UNIVERSITY, DEPARTMENT OF PSYCHOLOGY, 1975).

DESCRIPTION:

ONE HUNDRED SEVENTY-SIX NONPROGRAMMERS LEARNED A COMPUTER PROGRAMMING LANGUAGE BY A METHOD (MODEL) THAT EMPHASIZED A DIAGRAM MODEL OF A COMPUTER EXPRESSED IN FAMILIAR TERMS OR BY A METHOD WITH NO MODEL (NONMODEL) AND THEN PRACTICED ON EXERCISES AND TOOK A POST TEST. IN LEARNING AND POST TEST PERFORMANCE, MODEL SUBJECTS PERFORMED BEST ON INTERPRETATION OF PROGRAMS AND ON PROBLEMS REQUIRING LOOPING, WHILE NONMODEL SUBJECTS EXCELLED ON STRAIGHTFORWARD GENERATION OF PROGRAMS. THE MODEL WAS ESPECIALLY HELPFUL FOR LOW ABILITY SUBJECTS. PRACTICE IN INTERPRETATION HELPED NONMODEL SUBJECTS MOST AND PRACTICE IN WRITING SIMPLE PROGRAMS HELPED MODEL SUBJECTS MOST. THE ROLES OF THE MODEL IN ESTABLISHING A MEANINGFUL LEARNING SET, AND OF PRACTICE ON MATHAGENIC ACTIVITY, WERE DISCUSSED. (A)

10P, 16R.

245 PROGRAM COMPLEXITY

MCCABE, T.J. A COMPLEXITY MEASURE. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1976, SE-2, 308-320.

246 SOFTWARE ENGINEERING

MCCLURE, C.T. TOP-DOWN, BOTTOM-UP, AND STRUCTURED PROGRAMMING. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 397-402.

247 PROGRAMMING

MCCRACKEN, D.D. REVOLUTION IN PROGRAMMING: AN OVERVIEW. DATAMATION, DECEMBER 1973, 50-52.

DESCRIPTION:

THIS BRIEF ARTICLE INTRODUCES A SPECIAL ISSUE ON STRUCTURED PROGRAMMING. (HRR)

3P, OR.

248 AUTOMATIC PROGRAMMING

MCCUNE, B.P. THE PSI PROGRAM MODEL BUILDER. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 130-139 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 130-139).

DESCRIPTION:

A SYSTEM CALLED THE PROGRAM MODEL BUILDER (PMB) IS BEING DESIGNED AND IMPLEMENTED TO PERFORM THE BASIC OPERATIONS REQUIRED TO SYNTHESIZE AND MODIFY PROGRAMS. PMB PLAYS A CENTRAL ROLE AS ONE OF THE EXPERT MODULES OF THE PSI PROGRAM SYNTHESIS SYSTEM. PMB BUILDS A COMPLETE AND CONSISTENT PROGRAM MODEL FROM SMALL CHUNKS OF PROGRAM SPECIFICATION RECEIVED FROM OTHER PSI EXPERTS. PMB MUST DEAL WITH THE FACT THAT THESE PROGRAM FRAGMENTS OFTEN OMIT DETAILS AND MAY BE INCOMPLETE, AMBIGUOUS, INCONSISTENT, NONSPECIFIC, AND ARBITRARILY ORDERED. THE INITIAL VERSION OF PMB HAS SUCCESSFULLY SYNTHESIZED A FEW PROGRAM MODELS FROM FRAGMENTS. THIS WORK INCLUDES THE EVOLUTION OF A VERY HIGH-LEVEL PROGRAM MODELLING LANGUAGE, THE IDENTIFICATION AND CODIFICATION OF USEFUL VERY HIGH-LEVEL PROGRAMMING KNOWLEDGE INCLUDING EQUIVALENCE TRANSFORMATIONS, AND THE IMPLEMENTATION OF A RULE-BASED PROBLEM SOLVING SYSTEM EMBODYING THIS KNOWLEDGE. (A)

10P, 12R.

249 NATURAL-LANGUAGE PROGRAMMING

MCGEE, R.T. THE TRANSLATION OF DATA STRUCTURE REPRESENTATIONS OF SIMPLE QUEUING PROBLEMS INTO GPSS PROGRAMS AND ENGLISH TEXT. MONTEREY, CALIFORNIA: NAVAL POSTGRADUATE SCHOOL, JUNE 1971. (NTIS NO. AD 757701)

DESCRIPTION:

ONE OF THE GOALS OF COMPUTER TECHNOLOGY IS TO HAVE THE ABILITY TO COMMUNICATE WITH THE COMPUTER IN A NATURAL LANGUAGE SUCH AS ENGLISH. A RESEARCH EFFORT UNDERWAY AT THE NAVAL POSTGRADUATE SCHOOL INVOLVES THE DESIGN AND IMPLEMENTATION OF A COMPUTER SYSTEM FOR TRANSLATING NATURAL LANGUAGE DESCRIPTIONS OF SIMULATION PROBLEMS INTO EXECUTABLE COMPUTER PROGRAMS. IN THIS SYSTEM, ENGLISH TEXT IS TRANSLATED INTO AN INTERNAL DATA STRUCTURE WHICH IS THEN TRANSLATED INTO A COMPUTER PROGRAM FOR PERFORMING THE SIMULATION. THIS PAPER REPORTS ON AN EFFORT MADE TO AID THE USER OF THIS SYSTEM BY (1) EXTENDING THE CAPABILITIES OF AN EXISTING PROCEDURE FOR TRANSLATING THE INTERNAL DATA STRUCTURE INTO A GPSS SIMULATION PROGRAM, AND (2) DEVELOPING A PROCEDURE FOR TRANSLATING THE DATA STRUCTURE INTO ENGLISH TEXT SO THE USER COULD SEE THAT HIS INPUT TEXT HAD BEEN CORRECTLY INTERPRETED. THE BASIC OPERATION OF THE SYSTEM IS DESCRIBED AND EXAMPLES ARE GIVEN TO ILLUSTRATE THE SYSTEM'S CAPABILITIES. (A)

75P, .OR.

250 STRUCTURED PROGRAMMING
MCGOWAN, C. STRUCTURED PROGRAMMING: A REVIEW OF SOME PRACTICAL CONCEPTS. COMPUTER MAGAZINE, JUNE 1975, 8(6), 25-30.
DESCRIPTION:
THE SOFTWARE DEVELOPMENT PROCESS IS BEING REEXAMINED CRITICALLY THESE DAYS. THERE ARE AT LEAST TWO REASONS FOR THIS RENEWED SCRUTINY: FIRST, THE INEXORABLE ADVANCE OF HARDWARE TECHNOLOGY HAS NOW MADE SOFTWARE COSTS THE CLEARLY DOMINANT COMPONENT OF COMPUTING COSTS. SECOND, AMIDST THE FUROR OVER STRUCTURED PROGRAMMING (SP), WE HAVE COLLECTIVELY REALIZED THAT WE CAN INDEED DO BETTER (IN AT LEAST THE CODING SUBPROCESS OF SOFTWARE PRODUCTION). THIS REALIZATION LEADS US TO RECONSIDER CURRENT DESIGN AND IMPLEMENTATION STRATEGIES FOR POSSIBLY BETTER WAYS.
SP HAS BECOME A RALLYING POINT FOR METHODOLOGY CHANGES. IN THE LONG RUN, THIS MAY WELL BE ITS MOST SIGNIFICANT IMPACT UPON PRODUCTION PROGRAMMING PRACTICES. SEVERAL NEW SOFTWARE TECHNIQUES AND TOOLS HAVE GATHERED UNDER THE GONFALON OF SP. IN THIS PAPER, I WILL DISCUSS SOME IMPORTANT PRACTICAL FEATURES OF THE ENSEMBLE CALLED SP. (A)
6P, 11R.

251 PROGRAMMING
MCGOWAN, C.L., & KELLY, J.R. TOP-DOWN STRUCTURED PROGRAMMING TECHNIQUES. NEW YORK, NEW YORK: PETROCELLI/CHARTER, 1975.

252 STRUCTURED PROGRAMMING
MCHENRY, R. MEASURING PROGRAMMING IMPROVEMENT AT IBM-FSD. COMPUTER MAGAZINE, JUNE 1975, 8(6), 49.

253 PROGRAMMING LANGUAGES
MCKEEMAN, W.M. ON PREVENTING PROGRAMMING LANGUAGES FROM INTERFERING WITH PROGRAMMING. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 19-26.
DESCRIPTION:
WIRTH HAS PROPOSED A METHOD OF "STEPWISE REFINEMENT" FOR WRITING COMPUTER PROGRAMS. THIS PAPER PROPOSES THAT THE STEPS BE EXPRESSED AS PROOFS. A PROGRAM FOR THE EIGHT-QUEENS PROBLEM IS DEVELOPED, AND THE PROOF METHOD IS APPLIED ACROSS TWO OF THE STEPS OF THE DEVELOPMENT. THE STRENGTHS AND WEAKNESSES OF THE METHOD, AND ITS IMPLICATIONS FOR THE PROGRAMMING PROCESS AND PROGRAMMING LANGUAGE DESIGN ARE DISCUSSED. (A)
8P, 13R.

254 THROWAWAY MODULES, APL
MCLEAN, E.R. THE CONCEPT OF THROWAWAY CODE. DATAMATION, MARCH 1977, 23(3), PP. 139-140; 142; 144.

255 PETRI NETS FOR MODELING INTERACTIVE SYSTEMS

MELDMAN, J.A. A NEW TECHNIQUE FOR MODELING THE BEHAVIOR OF MAN-MACHINE INFORMATION SYSTEMS. SLOAN MANAGEMENT REVIEW, 1977, 18(3), 29-46.

DESCRIPTION:

A SERIOUS PROBLEM IN UNDERSTANDING OR DESIGNING MAN-MACHINE SYSTEMS IS THE LACK OF POWERFUL, FORMAL TECHNIQUES FOR MODELING, OR DESCRIBING MAN-MACHINE INTERACTIONS. THIS PAPER FOCUSES ON MAN-MACHINE INTERACTIONS IN MANAGEMENT INFORMATION SYSTEMS. A MANAGEMENT INFORMATION SYSTEM HAS FOUR CRUCIAL CHARACTERISTICS THAT COMPLICATE MODELING -- A LARGE NUMBER OF INTERACTING SUBSYSTEMS, HIGHLY PARALLEL BEHAVIOR, ASYNCHRONOUS COORDINATION OF SUBSYSTEMS, AND ALTERNATIVE BEHAVIOR OF SUBSYSTEMS. IT IS SUGGESTED THAT PETRI NETS OFFER A TECHNIQUE FOR MODELING THAT IS FORMAL AND EXPLICIT, HIGHLY MODULAR, AND COMPREHENSIVE AND CAN AID IN BETTER UNDERSTANDING MAN-MACHINE INTERACTIONS. (MEA)

18P, 21R.

256 SOFTWARE DEVELOPMENT

MERWIN, P.E. ESTIMATING SOFTWARE DEVELOPMENT SCHEDULES AND COSTS. IN PROCEEDINGS OF THE ACM DESIGN AUTOMATED WORKSHOP, 1972.

257 DEBUGGING

MICHARD, A. ANALYSE DU TRAVAIL DE DIAGNOSTIC D'ERREURS LOGIQUES DANS UN PROGRAMME FORTRAN (ANALYSIS OF THE WORK OF DIAGNOSIS OF LOGICAL ERRORS IN A FORTRAN PROGRAM) (REPORT NO. C.O. 7602-R48). LE CHESNAY, FRANCE: INSTITUT DE RECHERCHE D'INFORMATIQUE ET D'AUTOMATIQUE, 1975.

258 PROGRAMMING PRACTICES

MILLER, E.F., JR. A SYNOPSIS OF FOUR APPROACHES TO EVALUATION OF MODERN PROGRAMMING PRACTICE (REPORT NO. RP-6). LA JOLLA, CALIFORNIA: SCIENCE APPLICATIONS, INC., OCTOBER 1975.

DESCRIPTION:

INTERNATIONAL BUSINESS MACHINES HAS COMPLETED A SERIES OF REPORTS, RADC-TR-300, WHICH OUTLINE A NUMBER OF PROCEDURES WHICH COLLECTIVELY ARE CONCEIVED OF AS MODERN PROGRAMMING PRACTICES. THESE MODERN PROGRAMMING PRACTICES ARE DESIGNED TO IMPROVE THE PRODUCTION OF SOFTWARE BY REDUCING ERRORS, ENHANCING AMENABILITY TO CHANGE, AND IMPROVING PERFORMANCE OF INDIVIDUAL PROGRAMMERS, THUS REDUCING COSTS AND CONCOMITANTLY LOWERING TIME TO COMPLETE SOFTWARE PROJECTS. SINCE IMPLEMENTATION OF THESE PRACTICES IS COSTLY IN TERMS OF TRAINING, SETTING OF STANDARDS AND MONITORING, IT IS CONSIDERED PRUDENT TO EVALUATE HOW MODERN PRACTICES MIGHT BE COMPARED TO CONVENTIONAL PROGRAMMING PRACTICES IN ORDER TO ASSESS THE VALUE OF IMPLEMENTING THE PROGRAM ON A LARGE SCALE. THIS PAPER SUMMARIZES THE ANALYSES MADE BY FOUR INDEPENDENT CONSULTANTS ABOUT HOW SUCH EVALUATIONS MIGHT BE CONDUCTED. (A)

19P, 5R.

259 STRUCTURED PROGRAMMING

MILLER, E. F., JR. & LINDAMOOD, G. E. STRUCTURED PROGRAMMING: TOP-DOWN APPROACH. DATAMATION, DECEMBER 1973, 50-52.

DESCRIPTION:

STRUCTURED PROGRAMMING, IF THE CURRENT LEVEL OF INTEREST AND CONTROVERSY WITHIN THE COMPUTING COMMUNITY IS ANY MEASURE, IS AN IDEA WHOSE TIME HAS COME. IN A VERY GENERAL WAY, STRUCTURED PROGRAMMING IS A REFLECTION OF THE CONCERN WITH FORM AND THE INTERRELATIONSHIPS WHICH EXIST BETWEEN THE ATTRIBUTES OF A "GOOD" PROGRAM AND WHAT THE PROGRAM IS SUPPOSED TO DO. THUS, THE INTENSE INTEREST IN STRUCTURED PROGRAMMING MAY BE A MANIFESTATION OF A COMING MATURATION OF COMPUTING WHICH IS INTRINSICALLY A HUMAN ACTIVITY. (A, ABBR.)

3P, 11R.

260 PROGRAMMING LANGUAGES

MILLER, E.F., JR., & WASSERMAN, A.I. HIGH ORDER LANGUAGE EVALUATION PROJECT: FINAL REPORT (TECHNICAL REPORT NO. 341-78-523-110-SF). SAN FRANCISCO, CALIFORNIA: SCIENCE APPLICATIONS, INC., FEBRUARY 1977.

261 TESTING

MILLER, J.C., & MALONEY, C.J. SYSTEMATIC MISTAKE ANALYSIS OF DIGITAL COMPUTER PROGRAMS. COMMUNICATIONS OF THE ACM, 1963, 6, 58-63.

DESCRIPTION:

EFFECTIVE PROGRAM TESTING REQUIRES THAT EVERY PART OF THE PROGRAM BE CONSIDERED. THIS PAPER DESCRIBES A METHOD FOR DETERMINING THE NECESSARY TEST CASES AND FACILITATING THE LOCATION OF ERRORS. A PRINCIPAL COMPONENT OF THIS METHOD IS THE IDENTIFICATION OF BRANCHPOINTS THAT ARE AFFECTED BY INPUT DATA. (MEA)

6P, 11R.

262 GENERAL

MILLER, L.A. HARLAN MILLS ON "THE PSYCHOLOGY OF QUALITY" (REPORT NO. RC 3779). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, MAY 1973.

DESCRIPTION:

IN PREPARING FOR AN OPEN DISCUSSION PANEL ON THE PSYCHOLOGY OF QUALITY SCHEDULED FOR THE 1972 IBM SYSTEMS DEVELOPMENT DIVISION PROGRAMMING SYMPOSIUM (MARCH 26-29, WASHINGTON, D.C.), A NUMBER OF TENTATIVE SUGGESTION QUESTIONS PREPARED BY THE AUTHOR (AS MODERATOR) WERE SENT TO DR. HARLAN MILLS, AND OTHER PANEL MEMBERS, FOR THEIR EVALUATION PRIOR TO THE SYMPOSIUM. THE QUESTIONS, AND DR. MILLS' RESPONSES, ARE REPRODUCED VERBATIM IN THIS REPORT.

DR. MILLS' COMMENTS REFLECT HIS THINKING AND PROPOSALS FOR ALMOST ALL ASPECTS OF PROGRAMMING, RANGING FROM PROGRAMMER SELECTION AND MANAGEMENT TO THE PROGRAMMING WORK ENVIRONMENT. IN VIEW OF THE INCREASING INTEREST IN AND INFLUENCE OF DR. MILLS' IDEAS (E.G., CHIEF PROGRAMMER TEAM, TOP-DOWN PROGRAMMING), IT WAS BELIEVED USEFUL TO COMMUNICATE THIS INFORMATION TO A BROAD AUDIENCE.

A SECOND OBJECTIVE OF THIS DOCUMENTATION IS TO STIMULATE GENERAL INTEREST IN LOOKING AT THE PROBLEMS OF PROGRAMMING AND PROGRAM QUALITY FROM A BEHAVIORAL POINT OF VIEW.

THE READER SHOULD KEEP IN MIND THAT THE QUESTIONS, AND ANSWERS, WERE INTENDED TO EXPLORE A VARIETY OF POSSIBLE RELATIONS BETWEEN PSYCHOLOGICAL FACTORS OF PROGRAMMERS AND THE QUALITY OF PROGRAMS PRODUCED BY THEM (DEFINED PRIMARILY IN TERMS OF MINIMIZING BUGS AND PROVIDING FOR EASY EFFICIENT MAINTENANCE AND MODIFICATION OF CODES). THE PAPER IS NOT A TECHNICAL TREATMENT OF THE SUBJECT MATTER, DUE TO THE INFORMAL ATTITUDINAL NATURE OF THE INTERCHANGE. THUS, LITERATURE CITATIONS AND OTHER DOCUMENTATION ARE NOT INCLUDED. THE SPECIFIC QUESTIONS WERE ORGANIZED INTO THREE MAIN CONTENT AREAS: (1) COGNITIVE AND TRAIT ASPECTS OF INDIVIDUAL PROGRAMMERS, (2) MANAGEMENT AND ORGANIZATION OF PROGRAMMING GROUPS AND PROJECTS, AND (3) THE PROGRAMMING WORK ENVIRONMENT.

22P, OR.

263 PROGRAMMING AND QUERY LANGUAGE PROPERTIES

MILLER, L.A. PROGRAMMING BY NON-PROGRAMMERS. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1974, 6, 237-260 (ALSO: RESEARCH REPORT RC-4280, IBM WATSON RESEARCH CENTER, YORKTOWN HEIGHTS, NY, 1973).

DESCRIPTION:

NON-PROGRAMMERS WERE ASKED TO ORGANIZE NATURAL ENGLISH COMMANDS OF A LABORATORY PROGRAMMING LANGUAGE INTO PROGRAMS FOR SOLVING NAME-SORTING PROBLEMS. THE PROBLEMS DIFFERED IN THE SORT CONCEPT TO BE PROGRAMMED (CONJUNCTION VS. DISJUNCTION) AND IN THE FORM OF EXPRESSION OF THE LETTER TESTS TO BE MADE ON THE NAMES (AFFIRMATION VS. NEGATION).

PROGRAMMING PERFORMANCE WAS FOUND TO BE IMPAIRED WITH DISJUNCTIVE CONCEPTS AND WITH LETTER TESTS INVOLVING NEGATION. DIFFERENT CLASSES OF PROGRAM STRUCTURE WERE IDENTIFIED AND WERE ASSOCIATED WITH CERTAIN PROBLEM CONDITIONS AND ERROR MEASURES. AN INFLUENCE OF PRIOR EXPERIENCE WITH PROCEDURES ON PERFORMANCE WAS SUGGESTED. PROGRAM DEBUGGING AND TESTING PERFORMANCE WAS CHARACTERIZED. (A)

24P, 17R.

264 PROGRAMMING

MILLER, L.A. NAIVE PROGRAMMER PROBLEMS WITH SPECIFICATION OF TRANSFER-OF-CONTROL. AFIPS CONFERENCE PROCEEDINGS, 1975, 44, 657-663.

DESCRIPTION:

WE HAVE CONDUCTED A SERIES OF EXPERIMENTS CONCERNING THE PROGRAMMING PERFORMANCE OF PERSONS WITH NO PRIOR CONTACT WITH COMPUTERS OTHER THAN THE TRAINING RECEIVED IN THE EXPERIMENTAL SESSIONS. OUR OBJECTIVE IN THESE EXPERIMENTS IS TO IDENTIFY DESIGN PRINCIPLES FOR FACILITATING COMMUNICATION BETWEEN THE NAIVE USER, AS A PROBLEM SOLVER, AND A COMPUTER SYSTEM. WE VIEW PROGRAMMING AS A PROBLEM SOLVING ACTIVITY, AN INSTANCE OF WHAT GENERALLY MAY BE CALLED "PROCEDURE SPECIFICATION." WE BELIEVE IT POSSIBLE TO DESIGN COMPUTERS AS OPTIMAL PROBLEM-SOLVING TOOLS ONLY IF THE OPERATING CHARACTERISTICS OF THE PROBLEM SOLVERS ARE KNOWN AND TAKEN INTO ACCOUNT. CONSEQUENTLY, WE ARE SEEKING TO DISCOVER THE PROBLEMS AND PROCESSES INVOLVED IN HUMAN SPECIFICATION OF PROCEDURES, USING EXPERIMENTAL LABORATORY METHODS.

THE TOPICS COVERED ARE: (1) INITIAL STUDIES DEMONSTRATING THE FEASIBILITY OF INVESTIGATING PROGRAMMING IN THE LABORATORY AND SUGGESTING EXPRESSION OF TRANSFER-OF-CONTROL AS A LOCUS OF DIFFICULTY, (2) STUDIES COMPARING VARIOUS MEANS FOR EXPRESSING TRANSFER-OF-CONTROL, (3) RESULTS OF EXPERIMENTS USING OUR "PROCEDURE TABLE", AND (4) RESULTS OF ANALYSIS OF SPECIFICATIONS IN NATURAL LANGUAGE. (A, ABBR.)

7P, 2R.

265 PROGRAMMING LANGUAGES

MILLER, L.A. NATURAL LANGUAGE PROCEDURES: GUIDES FOR PROGRAMMING LANGUAGE DESIGN. IN PROCEEDINGS OF THE 6TH CONGRESS OF THE INTERNATIONAL ERGONOMICS ASSOCIATION. SANTA MONICA, CALIFORNIA: HUMAN FACTORS SOCIETY, 1976.

DESCRIPTION:

THE OBJECTIVE OF THIS PAPER IS TO FIND A COMMON SET OF MECHANISMS FOR EXPRESSING PROCESS INFORMATION AS PROCEDURES. THE UNDERLYING ASSUMPTIONS ARE THAT 1) THERE IS A COMMON SET OF MECHANISMS FOR COMMUNICATING PROCEDURAL INFORMATION, 2) A MAPPING OF PROCESS INFORMATION ONTO SYNTACTIC STRUCTURES MAY COMMUNICATE SPECIFIC PROCESS INFORMATION, AND 3) THE DECODING OF PROCESS INFORMATION IS VERB-DRIVEN. A MODEL IS PROPOSED THAT CONSISTS OF 1) A DOMAIN ENCYCLOPEDIA WITH RELEVANT ATTRIBUTES AND RELATIONS FOR THE ELEMENTS IN A PROCEDURAL DOMAIN, 2) PROCEDURAL POSTULATES TO GUIDE THE EXTRACTION AND EXECUTION OF INFORMATION, AND 3) VERB PROGRAMS GIVING TEMPORAL PROCESS MEANINGS. THE MODEL WAS EVALUATED IN THE DOMAIN OF KITCHEN RECIPES. (MEA)

30P, 3R.

266 SPECIFICATION OF PROCEDURES IN NATURAL LANGUAGE

MILLER, L.A., & BECKER, C.A. PROGRAMMING IN NATURAL ENGLISH (TECHNICAL REPORT NO. RC-5137). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, NOVEMBER 1974. (NTIS NO. AD A003923)

DESCRIPTION:

COLLEGE STUDENTS WERE ASKED TO TYPE DETAILED SPECIFICATIONS OF PROCEDURES IN THEIR NATURAL LANGUAGE (ENGLISH) AS SOLUTIONS FOR A SET OF SIX FILE MANIPULATION PROBLEMS. THE LANGUAGE PRODUCTIONS WERE EXAMINED FROM THE POINTS OF VIEW OF SOLUTION CORRECTNESS, PREFERENCES OF EXPRESSIONS, CONTEXTUAL REFERENCING, WORD USAGE, AND FORMAL PROGRAMMING LANGUAGES. (A)

AMONG OTHER RESULTS, SOLUTIONS WERE GENERALLY SATISFACTORY ONLY FOR THE SIMPLEST PROBLEMS; FOR MORE COMPLEX PROBLEMS, SOLUTIONS TENDED TO BE INCOMPLETE. SUBJECTS TENDED TO SELECT THE SIMPLEST OF THE AVAILABLE ALGORITHMS; THESE WERE NOT NECESSARILY THE MOST EFFICIENT OR THE LEAST ERROR-PRONE APPROACHES. FORTY-TWO PERCENT OF THE DATA REFERENCES WERE CONTEXT-DEPENDENT. SUBJECTS TENDED TO TREAT DATA AGGREGATES, RATHER THAN INDIVIDUAL DATA ELEMENTS. LITTLE EXPLICIT TRANSFER OF CONTROL OCCURRED; THE PROCEDURES WERE MOSTLY LINEAR. (HRR)

58P, 38R.

267 GENERAL DISCUSSION OF ISSUES IN MAN-COMPUTER INTERACTION

MILLER, L.A., & THOMAS, J.C., JR. BEHAVIORAL ISSUES IN THE USE OF INTERACTIVE SYSTEMS. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1977, 9, 509-536 (ALSO TECHNICAL REPORT NO. RC-6326, YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, DECEMBER 1976).

DESCRIPTION:

THIS PAPER IDENTIFIES BEHAVIORAL ISSUES RELATED TO THE USE OF INTERACTIVE COMPUTERS PRIMARILY BY PERSONS WHO ARE NOT COMPUTER PROFESSIONALS, SO-CALLED "GENERAL USERS." THIS IS NOT AN EXHAUSTIVE LITERATURE SURVEY, BUT INSTEAD PROVIDES: (1) A STRUCTURE FOR DISCUSSING ISSUES OF INTERACTIVE COMPUTING, AND (2) THE AUTHORS' BEST ESTIMATE OF IMPORTANT BEHAVIORAL PROBLEMS, WITH SUGGESTIONS FOR SOLUTIONS.

THE DISCUSSION IS LIMITED IN THIS PAPER TO GENERAL ISSUES WHICH DO NOT TAKE INTO ACCOUNT THE USER'S PARTICULAR TASK. THE TWO MAJOR TOPICS ARE SYSTEM CHARACTERISTICS (PERFORMANCE, FACILITIES, AND ON-LINE INFORMATION), AND INTERFACE CHARACTERISTICS (DIALOGUE SYTLE, DISPLAYS AND GRAPHICS, OTHER INPUT/OUTPUT MEDIA). (A)

28P, 142R.

268 CHIEF PROGRAMMER TEAMS

MILLS, H.D. CHIEF PROGRAMMER TEAMS: TECHNIQUES AND PROCEDURES (IBM INTERNAL REPORT), JANUARY 1970.

269 STRUCTURED PROGRAMMING

MILLS, H.D. TOP-DOWN PROGRAMMING IN LARGE SYSTEMS. IN R. RUSTIN (ED.), DEBUGGING TECHNIQUES IN LARGE SYSTEMS. ENGLEWOOD CLIFFS, NEW JERSEY: PRENTICE-HALL, 1971, 41-55.

DESCRIPTION:

STRUCTURED PROGRAMMING CAN BE USED TO DEVELOP A LARGE SYSTEM IN AN EVOLVING TREE STRUCTURE OF NESTED PROGRAM MODULES, WITH NO CONTROL BRANCHING BETWEEN MODULES EXCEPT FOR MODULE CALLS DEFINED IN THE TREE STRUCTURE. BY LIMITING THE SIZE AND COMPLEXITY OF MODULES, UNIT DEBUGGING CAN BE DONE BY SYSTEMATIC READING, AND THE MODULES EXECUTED DIRECTLY IN THE EVOLVING SYSTEM IN A TOP DOWN TESTING PROCESS. (A)

15P, 15R.

270 SOFTWARE ENGINEERING

MILLS, H.). HOW TO WRITE CORRECT PROGRAMS AND KNOW IT. IN PROCEEDINGS, INTERNATIONAL CONFERENCE ON RELIABLE SOFTWARE, 21-23 APRIL 1975, LOS ANGELES, CALIFORNIA. SIGPLAN NOTICES, 1975, 10, 363-370 (ALSO IBM TECHNICAL REPORT NO. FSC 73-5008, GAITHERSBURG, MARYLAND: IBM CORP., 1973).

DESCRIPTION:

THERE IS NO FOOLPROOF WAY TO EVER KNOW THAT YOU HAVE FOUND THE LAST ERROR IN A PROGRAM. SO THE BEST WAY TO ACQUIRE CONFIDENCE THAT A PROGRAM HAS NO ERRORS IS NEVER TO FIND THE FIRST ONE, NO MATTER HOW MUCH IT IS TESTED AND USED. IT IS AN OLD MYTH THAT PROGRAMMING MUST BE AN ERROR-PRONE, CUT-AND-TRY PROCESS OF FRUSTRATION AND ANXIETY. THE NEW REALITY IS THAT YOU CAN LEARN TO CONSISTENTLY WRITE PROGRAMS WHICH ARE ERROR FREE IN THEIR DEBUGGING AND SUBSEQUENT USE. THIS NEW REALITY IS FOUNDED IN THE IDEAS OF STRUCTURED PROGRAMMING AND PROGRAM CORRECTNESS, WHICH NOT ONLY PROVIDE A SYSTEMATIC APPROACH TO PROGRAMMING, BUT ALSO MOTIVATE A HIGH DEGREE OF CONCENTRATION AND PRECISION IN THE CODING SUBPROCESS. (A)

271 SOFTWARE ENGINEERING
MILLS, H.D. SOFTWARE DEVELOPMENT. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1976, SE-2, 265-273.
DESCRIPTION:
SOFTWARE DEVELOPMENT HAS EMERGED AS A CRITICAL BOTTLENECK IN THE HUMAN USE OF AUTOMATIC DATA PROCESSING. BEGINNING WITH AD HOC HEURISTIC METHODS OF DESIGN AND IMPLEMENTATION OF SOFTWARE SYSTEMS, PROBLEMS OF SOFTWARE MAINTENANCE AND CHANGES HAVE BECOME UNEXPECTEDLY LARGE. IT IS CONTENDED THAT IMPROVEMENT IS POSSIBLE ONLY WITH MORE RIGOR IN SOFTWARE DESIGN AND DEVELOPMENT METHODOLOGY. RIGOROUS SOFTWARE DESIGN SHOULD SURVIVE ITS IMPLEMENTATION AND BE THE BASIS FOR FURTHER EVOLUTION. SOFTWARE DEVELOPMENT SHOULD BE DONE INCREMENTALLY, IN STAGES WITH CONTINUOUS USER PARTICIPATION AND REPLANNING, AND WITH DESIGN-TO-COST PROGRAMMING WITHIN EACH STAGE. (A)
9P, 18R.

272 SOFTWARE ENGINEERING
MILLS, H.D. SOFTWARE ENGINEERING. SCIENCE, 1977, 195, 1199-1205.
DESCRIPTION:
THE PRACTICAL CONTROL OF COMPUTERS AND THEIR VERY COMPLEXITY REQUIRES A MATHEMATICAL BASIS FOR THEIR UNDERSTANDING. SOFTWARE IS BETTER DEFINED AS THE "LOGICAL DOCTRINE FOR THE HARMONIOUS COOPERATION OF PEOPLE AND MACHINES." CURRENT DATA PROCESSING SYSTEMS ARE WORKING WELL ENOUGH TO BE INDISPENSABLE, BUT POORLY ENOUGH TO BE THE CAUSE OF UNTOLD FRUSTRATION. SOFTWARE ENGINEERING IS, THEREFORE, AN EMERGING IDEA. THREE OF ITS CURRENT AREAS OF DISCIPLINE AND STUDY ARE DISCUSSED. THESE ARE: THE DESIGN AND VERIFICATION OF SEQUENTIAL PROCESSES, THE INTERACTION OF PARALLEL OR "INDEPENDENT" PROCESSES, AND THE ORGANIZATION OF PROCESSES INTO SYSTEMS OF ABSTRACT MACHINES. (GDC)

273 USER REQUIREMENTS ANALYSIS
MJOSUND, A. TOWARD A STRATEGY FOR INFORMATION NEEDS ANALYSIS. COMPUTERS AND OPERATIONS RESEARCH, 1975, 2, 39-47.
DESCRIPTION:
INFORMATION NEEDS ANALYSIS IS A PREREQUISITE FOR DESIGN OF AN EFFECTIVE INFORMATION SYSTEM. HOWEVER, THE PROBLEMS ASSOCIATED WITH SUCH ANALYSIS HAVE BEEN LARGELY NEGLECTED IN THE INFORMATION SYSTEMS LITERATURE. TWO SIMULTANEOUS GENERAL APPROACHES ARE SUGGESTED WHICH ARE EXPECTED TO CONTRIBUTE TO THE SOLUTION OF THESE PROBLEMS. ONE IS TO USE THE INFORMATION SYSTEM ANALYSIS TO GUIDE RESEARCH, OR APPLICATION OF RESULTS FROM RESEARCH, TO SOLVE MANAGEMENT PROBLEMS. THE OTHER IS TO FOLLOW A STRATEGY IN THE ANALYSIS OF INFORMATION NEEDS SUCH THAT THE STEPS IN THIS ANALYSIS ARE CLOSELY RELATED TO THE STRUCTURE RELATING THE DECISIONS AND ACTIONS IN THE ORGANIZATION. A GROSS CLASSIFICATION SCHEME IS PROPOSED TO AID IN DETERMINING THE STRATEGY. (A)
9P, 7R.

274 MAN-COMPUTER DIALOGUE
MOORE, R.K., & MAIN, W. INTERACTIVE LANGUAGES: DESIGN CRITERIA AND A PROPOSAL. AFIPS CONFERENCE PROCEEDINGS, 1968, 33 (PT. 1), 193-200.
DESCRIPTION:
THIS PAPER DISCUSSES THE DESIGN CRITERIA FOR INTERACTIVE PROGRAMMING LANGUAGES AND DESCRIBES TCL (TYMSHARE CONVERSATIONAL LANGUAGE). THE SALIENT FEATURES OF TCL INCLUDE LONG VARIABLE NAMES, SYMBOLIC STATEMENT LABELS, FULL-FLEDGED SUBPROGRAMS WITH PARAMETERS, AND THE ABILITY TO HANDLE RECURSIVE PROCEDURAL ALGORITHMS. (MEA)
8P, 2R.

275 PROGRAMMING

MORGAN, H.L. SPELLING CORRECTIONS IN SYSTEMS PROGRAMS. COMMUNICATIONS OF THE ACM, 1970, 13, 90-94.

DESCRIPTION:

SEVERAL SPECIALIZED TECHNIQUES ARE SHOWN FOR EFFICIENTLY INCORPORATING SPELLING CORRECTION ALGORITHMS INTO COMPILERS AND OPERATING SYSTEMS. THESE INCLUDE THE USE OF SYNTAX AND SEMANTICS INFORMATION, THE ORGANIZATION OF RESTRICTED KEYWORD AND SYMBOL TABLES, AND THE CONSIDERATION OF A LIMITED CLASS OF SPELLING ERRORS. SAMPLE 360 CODING FOR PERFORMING SPELLING CORRECTION IS PRESENTED. BY USING SYSTEMS WHICH PERFORM SPELLING CORRECTION, THE NUMBER OF DEBUGGING RUNS PER PROGRAM HAS BEEN DECREASED, SAVING BOTH PROGRAMMER AND MACHINE TIME.

5P, 11R.

276 SOFTWARE DESIGN

MORSENSTERN, M. AUTOMATING THE SOFTWARE DESIGN PROCESS FOR MANAGEMENT INFORMATION SYSTEMS. IN PROCEEDINGS OF THE COMPUTER SOFTWARE AND APPLICATIONS CONFERENCE. NEW YORK: INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC., 1977, 642-648.

DESCRIPTION:

AN OPERATIONAL PROTOTYPE FACILITY HAS BEEN DEVELOPED WHICH AUTOMATES THE DESIGN OF SOFTWARE FOR BATCH-ORIENTED MANAGEMENT INFORMATION SYSTEMS. THE GLOBAL OPTIMIZATION CONSIDERATIONS INCLUDE THE DESIGN OF THE FILE SYSTEM, STRUCTURING OF EACH RUN, INTER-RUN DATA FLOW, ACCESS METHODS, FILE ORGANIZATIONS, AND SORTING. AN ANALYSIS OF THE INTERDEPENDENCIES ENABLES US TO ACCOUNT FOR THE NON-LOCAL EFFECTS OF THE DESIGN DECISIONS, SUCH AS THE INTERACTIONS WHICH OCCUR AMONG THE SORT ORDERS OF MULTIPLE KEY FILES AND THE SELECTION OF BLOCKING FACTORS. BOTH PRACTICAL AND THEORETICAL EVALUATION OF THIS DESIGNER-OPTIMIZER FACILITY INDICATES THAT THE RESULTING DESIGNS ARE GOOD, AND ARE COMPARABLE TO THOSE LIKELY TO BE PRODUCED BY A SYSTEM DESIGNER WHO IS LIMITED TO THE SAME REPERTOIRE OF TECHNIQUES. (A)

7P, 5R.

277 PROGRAMMING LANGUAGES

MOULTON, P.G., & MULLER, M.E. DITRAN: A COMPILER EMPHASIZING DIAGNOSTICS. COMMUNICATIONS OF THE ACM, 1967, 10, 45-52.

DESCRIPTION:

DITRAN (DIAGNOSTIC FORTRAN) IS AN IMPLEMENTATION OF NSA BASIC FORTRAN WITH RATHER EXTENSIVE ERROR CHECKING CAPABILITIES BOTH AT COMPILATION TIME AND DURING EXECUTION OF A PROGRAM. THE NEED FOR IMPROVED DIAGNOSTIC CAPABILITIES AND SOME OBJECTIVES TO BE MET BY ANY COMPILER ARE DISCUSSED. ATTENTION IS GIVEN TO THE DESIGN AND IMPLEMENTAION OF DITRAN AND THE PARTICULAR TECHNIQUES EMPLOYED TO PROVIDE THE DIAGNOSTIC FEATURES. THE HANDLING OF ERROR MESSAGES BY A GENERAL MACRO APPROACH IS DESCRIBED. SPECIAL FEATURES WHICH PROVIDE TEACHING AIDS FOR USE BY INSTRUCTORS ARE NOTED. (A)

8P, 16R.

278 SOFTWARE RELIABILITY

MUSA, J.D. A THEORY OF SOFTWARE RELIABILITY AND ITS APPLICATION. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 312-327.

DESCRIPTION:

AN APPROACH TO A THEORY OF SOFTWARE RELIABILITY BASED ON EXECUTION TIME IS DERIVED. THIS APPROACH PROVIDES A MODEL THAT IS SIMPLE, INTUITIVELY APPEALING, AND IMMEDIATELY USEFUL.

THE THEORY PERMITS THE ESTIMATION, IN ADVANCE OF A PROJECT, OF THE AMOUNT OF TESTING IN TERMS OF EXECUTION TIME REQUIRED TO ACHIEVE A SPECIFIED RELIABILITY GOAL (STATED AS A MEAN TIME TO FAILURE (MTTF)). EXECUTION TIME CAN THEN BE RELATED TO CALENDAR TIME, PERMITTING A SCHEDULE TO BE DEVELOPED. ESTIMATES OF EXECUTION TIME AND CALENDAR TIME REMAINING UNTIL THE RELIABILITY GOAL IS ATTAINED CAN BE CONTINUALLY REMADE AS TESTING PROCEEDS, BASED ONLY ON THE LENGTH OF THE EXECUTION TIME INTERVALS BETWEEN FAILURES. THE CURRENT MTTF AND THE NUMBER OF ERRORS REMAINING CAN ALSO BE ESTIMATED. MAXIMUM LIKELIHOOD ESTIMATION IS EMPLOYED, AND CONFIDENCE INTERVALS ARE ALSO ESTABLISHED. THE FOREGOING INFORMATION IS OBVIOUSLY VERY VALUABLE IN SCHEDULING AND MONITORING THE PROGRESS OF PROGRAM TESTING. A PROGRAM HAS BEEN IMPLEMENTED TO COMPUTE THE FOREGOING QUANTITIES.

THE RELIABILITY MODEL THAT HAS BEEN DEVELOPED CAN BE USED IN MAKING SYSTEM TRADEOFFS INVOLVING SOFTWARE OR SOFTWARE AND HARDWARE COMPONENTS. IT ALSO PROVIDES A SOUNDLY BASED UNIT OF MEASURE FOR THE COMPARATIVE EVALUATION OF VARIOUS PROGRAMMING TECHNIQUES THAT ARE EXPECTED TO ENHANCE RELIABILITY.

THE MODEL HAS BEEN APPLIED TO FOUR MEDIUM-SIZED SOFTWARE DEVELOPMENT PROJECTS, ALL OF WHICH HAVE COMPLETED THEIR LIFE CYCLES. MEASUREMENTS TAKEN OF MTTF DURING OPERATION AGREE WELL WITH THE PREDICTIONS MADE AT THE END OF SYSTEM TEST. AS FAR AS THE AUTHOR CAN DETERMINE, THESE ARE THE FIRST TIMES THAT A SOFTWARE RELIABILITY MODEL HAS BEEN USED DURING SOFTWARE DEVELOPMENT PROJECTS. THE PAPER REFLECTS AND INCORPORATES THE PRACTICAL EXPERIENCE GAINED. (A)

16P, 20R.

279 SOFTWARE DESIGN

MYERS, G. J. RELIABLE SOFTWARE THROUGH COMPOSITE DESIGN. NEW YORK: PETROCELLI, 1975.

280 SOFTWARE RELIABILITY

MYERS, G.J. SOFTWARE RELIABILITY: PRINCIPLES AND PRACTICES. NEW YORK: JOHN WILEY AND SONS, 1976.

281 PROGRAM COMPLEXITY

MYERS, G.J. AN EXTENSION TO THE CYCLOMATIC MEASURE OF PROGRAM COMPLEXITY. SIGPLAN NOTICES, OCTOBER 1977, 12(10), 61-64.

DESCRIPTION:

A RECENT PAPER HAS DESCRIBED A GRAPH-THEORETIC MEASURE OF PROGRAM COMPLEXITY, WHERE A PROGRAM'S COMPLEXITY IS ASSUMED TO BE ONLY A FACTOR OF THE PROGRAM'S DECISION STRUCTURE. HOWEVER, SEVERAL ANOMALIES HAVE BEEN FOUND WHERE A HIGHER COMPLEXITY MEASURE WOULD BE CALCULATED FOR A PROGRAM OF LESSER COMPLEXITY THAN FOR A MORE-COMPLEX PROGRAM. THIS PAPER DISCUSSES THESE ANOMALIES, DESCRIBES A SIMPLE EXTENSION TO THE MEASURE TO ELIMINATE THEM, AND APPLIES THE MEASURE TO SEVERAL PROGRAMS IN THE LITERATURE. (A)

4P, 3R.

282 PROGRAMMING

NAGY, G., & PENNEBAKER, M.C. A STEP TOWARD AUTOMATIC ANALYSIS OF STUDENT PROGRAMMING ERRORS IN A BATCH ENVIRONMENT. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1974, 6, 563-578.

DESCRIPTION:

THE OBJECT OF THIS INVESTIGATION IS TO DEVELOP A METHOD FOR THE AUTOMATIC COLLECTION OF MEANINGFUL STATISTICAL INFORMATION ABOUT THE CAUSES OF PROGRAM RESUBMITTAL IN A BATCH-PROCESSING ENVIRONMENT. SUCCESSIVE VERSIONS OF A GIVEN PROGRAM ARE COMPARED STATEMENT-BY-STATEMENT IN ORDER TO ISOLATE MINOR CHANGES MADE IN THE PROGRAM. ALL STATEMENTS INSERTED, SUBSTITUTED, OR DELETED ARE EXAMINED IN TERMS OF (1) THE TYPE OF STATEMENT (I.E. DO, IF, ETC.), (2) THE NUMBER OF CONSECUTIVE STATEMENTS INVOLVED IN THE CHANGE, AND (3) HOW MANY TIMES THE PROGRAM HAS ALREADY BEEN SUBMITTED (THE NUMBER OF "TRIES"). ONE THOUSAND ONE HUNDRED AND TEN PROGRAMS ARE ANALYZED IN THIS MANNER. THE METHOD IS INTENDED TO BE USED, IN CONJUNCTION WITH DETAILED STUDY OF SELECTED CASES AND WITH FURTHER EXPERIMENTATION IN COMPLETELY CONTROLLED SITUATIONS, TO IMPROVE PROGRAMMING INSTRUCTION AND MANUALS, TO PRODUCE BETTER DIAGNOSTIC MESSAGES, TO AID IN THE DESIGN OF NEW COMPILERS, AND EVENTUALLY TO PROVIDE FOR AUTOMATIC CORRECTION OF TRIVIAL MISTAKES. (A) 16P, 14R.

283 SOFTWARE ENGINEERING

NAUR, P. PROGRAMMING BY ACTION CLUSTERS. BIT, 1969, 9, 250-258.

284 SOFTWARE DEVELOPMENT

NAUR, P. AN EXPERIMENT ON PROGRAM DEVELOPMENT. BIT, 1972, 12, [illegible]-365.

DESCRIPTION:

AS A CONTRIBUTION TO PROGRAMMING METHODOLOGY, THE PAPER CONTAINS A DETAILED, STEP-BY-STEP ACCOUNT OF THE CONSIDERATIONS LEADING TO A PROGRAM FOR SOLVING THE 8-QUEENS PROBLEM. THE EXPERIENCE IS RELATED TO THE METHOD OF STEPWISE REFINEMENT AND TO GENERAL PROBLEM SOLVING TECHNIQUES. (A)

285 SOFTWARE ENGINEERING

NAUR, P., RANDELL, B., & BUXTON, J.N. (EDS.). SOFTWARE ENGINEERING: CONCEPTS AND TECHNIQUES. PROCEEDINGS OF THE NATO CONFERENCES. NEW YORK: PETROCELLI/CHARTER, 1976.

DESCRIPTION:

THE PRESENT REPORT IS CONCERNED WITH A PROBLEM CRUCIAL TO THE USE OF COMPUTERS, VIZ. THE SO-CALLED SOFTWARE, OR PROGRAMS, DEVELOPED TO CONTROL THEIR ACTION. THE REPORT SUMMARISES THE DISCUSSIONS AT A WORKING CONFERENCE ON SOFTWARE ENGINEERING, SPONSORED BY THE NATO SCIENCE COMMITTEE. THE CONFERENCE WAS ATTENDED BY MORE THAN FIFTY PEOPLE, FROM ELEVEN DIFFERENT COUNTRIES, ALL CONCERNED PROFESSIONALLY WITH SOFTWARE, EITHER AS USERS, MANUFACTURERS, OR TEACHERS AT UNIVERSITIES. THE DISCUSSIONS COVER ALL ASPECTS OF SOFTWARE INCLUDING: (1) RELATION OF SOFTWARE TO THE HARDWARE OF COMPUTERS; (2) DESIGN OF SOFTWARE; (3) PRODUCTION, OR IMPLEMENTATION OF SOFTWARE; (4) DISTRIBUTION OF SOFTWARE; AND (5) SERVICE ON SOFTWARE.

BY INCLUDING MANY DIRECT QUOTATIONS AND EXCHANGES OF OPINION, THE REPORT REFLECTS THE LIVELY CONTROVERSIES OF THE ORIGINAL DISCUSSION.

ALTHOUGH MUCH OF THE DISCUSSIONS WERE OF A DETAILED TECHNICAL NATURE, THE REPORT ALSO CONTAINS SECTIONS REPORTING ON DISCUSSIONS WHICH WILL BE OF INTEREST TO A MUCH WIDER AUDIENCE. THIS HOLDS FOR SUBJECTS LIKE: (1) THE PROBLEMS OF ACHIEVING SUFFICIENT RELIABILITY IN THE DATA SYSTEMS WHICH ARE BECOMING INCREASINGLY INTEGRATED INTO THE CENTRAL ACTIVITIES OF MODERN SOCIETY; (2) THE DIFFICULTIES OF MEETING SCHEDULES AND SPECIFICATIONS ON LARGE SOFTWARE PROJECTS; (3) EDUCATION OF SOFTWARE (OR DATA SYSTEMS) ENGINEERS; AND (4) THE HIGHLY CONTROVERSIAL QUESTION OF WHETHER SOFTWARE SHOULD BE PRICED SEPARATELY FROM HARDWARE.

THUS, WHILE THE REPORT IS OF PARTICULAR CONCERN TO THE IMMEDIATE USERS OF COMPUTERS AND TO COMPUTER MANUFACTURERS, MANY POINTS MAY SERVE TO ENLIGHTEN AND WARN POLICY MAKERS AT ALL LEVELS. READERS FROM THE WIDER AUDIENCE SHOULD NOTE, HOWEVER, THAT THE CONFERENCE WAS CONCENTRATING ON THE BASIC ISSUES AND KEY PROBLEMS IN THE CRITICAL AREAS OF SOFTWARE ENGINEERING. IT THEREFORE DID NOT ATTEMPT TO PROVIDE A BALANCED REVIEW OF THE TOTAL STATE OF SOFTWARE, AND TENDS TO UNDERSTRESS THE ACHIEVEMENTS OF THE FIELD. (A)

286 STRUCTURED PROGRAMMING

NEELY, P.M. THE NEW PROGRAMMING DISCIPLINE. SOFTWARE: PRACTICE AND EXPERIENCE, 1976, 6, 7-27.

DESCRIPTION:

RECENTLY THERE HAS BEEN SUBSTANTIAL INTEREST IN PROMOTING "STRUCTURED PROGRAMMING" AS A MEANS OF WRITING MORE NEARLY ERROR FREE PROGRAMS. HOWEVER SINCE THE CHIEF ADVOCATES OF STRUCTURED PROGRAMMING USE ALGOL OR PASCAL, AND DISDAIN FORTRAN, THERE IS A DIFFICULTY IN COMMUNICATION. SINCE IT IS MY PERCEPTION THAT STRUCTURED PROGRAMMING AND THE LESSONS TO BE LEARNED FROM PROOFS OF CORRECTNESS CAN BE APPLIED IN ANY LANGUAGE, INCLUDING FORTRAN, I FEEL THAT THESE IDEAS SHOULD BE PROMULGATED TO APPLIED SCIENTIFIC PROGRAMMERS.

HENCE THIS PAPER WILL COMMENCE WITH A SUMMARY OF THE WHOLE COMPLEX OF IDEAS AND PRACTICES THAT ARE SUBSUMED UNDER THE TERM "STRUCTURED PROGRAMMING". THEN SOME SIMPLE EXAMPLES OF TOP DOWN DESIGN AND PROGRAMMING WILL BE GIVEN. FINALLY I WILL RETURN TO A DISCUSSION OF SOME OF THE PROBLEMS WHICH ARE LIKELY TO BE ENCOUNTERED IN THE USE AND PROMULGATION OF STRUCTURED PROGRAMMING. (A)

21P, 7R.

287 PROGRAMMING

NEWSTED, P.R. FORTRAN PROGRAM COMPREHENSION AS A FUNCTION OF DOCUMENTATION. MILWAUKEE, WISCONSIN: UNIVERSITY OF WISCONSIN, SCHOOL OF BUSINESS ADMINISTRATION, UNDATED.

DESCRIPTION:

BEHAVIORAL DATA ARE PRESENTED WHICH INDICATE THAT COMMENTS AND MNEMONIC VARIABLE NAMES MAY NOT ALWAYS IMPROVE STUDENT COMPREHENSION OF FORTRAN PROGRAMS. INTERACTION OF THESE VARIABLES WITH PROGRAM DIFFICULTY SUGGESTS THAT THEY ARE USEFUL ONLY AFTER A GIVEN DIFFICULTY LEVEL IS REACHED -- A LEVEL BEYOND WHICH IT IS NOT POSSIBLE TO CONCEPTUALIZE A PROGRAM AS A SINGLE IDEA. (A)

TWO EXPERIMENTS WERE PERFORMED: (1) A BETWEEN-GROUPS EXPERIMENT USING A SINGLE FORTRAN PROGRAM AND VARYING COMMENTS (PRESENT OR ABSENT) AND VARIABLE NAMES ("MNEMONIC" OR "NON-MNEMONIC"), AND (2) A WITHIN-SUBJECT EXPERIMENT USING FOUR PROGRAMS WHICH VARIED IN DIFFICULTY ("EASY", "HARD") AND VARIABLE TYPE ("MNEMONIC", "NON-MNEMONIC"). AFTER EXPOSURE TO EACH PROGRAM, SUBJECTS WERE GIVEN MULTIPLE-CHOICE COMPREHENSION TESTS. NO SIGNIFICANT EFFECTS WERE FOUND IN THE FIRST EXPERIMENT. IN THE SECOND, BOTH MAIN EFFECTS (DIFFICULTY, VARIABLE TYPE) AND THEIR INTERACTION WERE SIGNIFICANT. THE "EASY, NON-MNEMONIC" PROGRAM WAS ASSOCIATED WITH BETTER TEST PERFORMANCE THAN THE "EASY, MNEMONIC" PROGRAM. (HRR)

7P, 6R.

288 PROGRAMMING

NEWSTED, P. GRADE AND ABILITY PREDICTIONS IN AN INTRODUCTORY PROGRAMMING COURSE (TECHNICAL REPORT). MILWAUKEE, WISCONSIN: UNIVERSITY OF WISCONSIN, SCHOOL OF BUSINESS ADMINISTRATION, 1974.

289 REVIEW OF USER INTERACTION WITH NAVY COMPUTER SYSTEMS

NICHOLSON, R.M., WIGGINS, B.D., & SILVER, C.A. AN INVESTIGATION INTO SOFTWARE STRUCTURES FOR MAN/MACHINE INTERACTIONS. ARLINGTON, VIRGINIA: ANALYTICS, INC., FEBRUARY 1972. (NTIS NO. AD 737266)

DESCRIPTION:

THE CURRENT TREND IN COMMAND AND CONTROL/INFORMATION SYSTEMS WITHIN THE NAVY, TOWARD GREATER USE OF INTERACTIVE CAPABILITIES, HAS THE EFFECT OF BRINGING THE TRUE "USER" -- THE DECISION MAKER -- INTO DIRECT CONTACT WITH THE SYSTEM, RATHER THAN USING A PROGRAMMER AS AN INTERMEDIARY. IT IS THEREFORE NECESSARY THAT THE SYSTEM DESIGNER ORIENT THE MAN/MACHINE COMMUNICATION LESS TOWARD HIS OWN PROGRAMMING COMMUNITY AND MORE TOWARD A USER WHOSE FAMILIARITY WITH COMPUTER DEVICES AND TERMINOLOGY IS SOMEWHAT LESS THAN HIS OWN.

FOR A CLEAR VIEW OF THE TYPICAL USER AND THE FUNCTIONS HE AND THE SYSTEM PERFORM, A SURVEY OF RECENT NAVY SYSTEMS IS DESCRIBED. A REVIEW OF THE LITERATURE IN INFORMATION SYSTEMS TO DETERMINE THE AVAILABILITY OF INFORMATION USEFUL TO THE SYSTEM DESIGNER IN INTERACTIVE SOFTWARE PERFORMANCE IS PRESENTED. FINALLY, A RESEARCH PROGRAM TO DERIVE THE NEEDED INFORMATION IS PROPOSED. (A)

92P, 55R.

290 PROGRAMMING

NICKERSON, R.S. SOME COMMENTS ON SOFTWARE DEVELOPMENT. IN PROCEEDINGS, ARMY HUMAN FACTORS RESEARCH AND DEVELOPMENT CONFERENCE, OCTOBER 1970.

291 GENERAL DISCUSSION OF HUMAN FACTORS IN COMPUTER SYSTEMS

NICKERSON, R.S., ELKIND, J.I., & CARBONELL, J.R. HUMAN FACTORS AND THE DESIGN OF TIME SHARING COMPUTER SYSTEMS. HUMAN FACTORS, 1968, 10, 127-133.

DESCRIPTION:

THE ADVENT OF COMPUTER TIME SHARING POSES AN EXTRAORDINARY CHALLENGE TO HUMAN FACTORS RESEARCH DURING THE NEXT DECADE. BEFORE TIME SHARING, TWO FACTS COMBINED TO DE-EMPHASIZE THE IMPORTANCE OF HUMAN FACTORS CONSIDERATIONS IN THE DESIGN OF COMPUTER SYSTEMS: (1) THE COST OF THE COMPUTER'S TIME WAS EXORBITANTLY HIGH RELATIVE TO THE COST OF USERS' TIME, AND (2) THE USERS CONSTITUTED A SELECT, HIGHLY SKILLED AND HIGHLY MOTIVATED GROUP OF SPECIALISTS. TWO OF THE PROMISES OF TIME SHARING, HOWEVER, ARE (1) A DRASTIC REDUCTION IN THE COST OF COMPUTER TIME TO THE INDIVIDUAL USER, AND (2) THE LARGE SCALE AVAILABILITY OF COMPUTER FACILITIES TO INDIVIDUALS UNTRAINED IN ANY AREAS OF COMPUTER TECHNOLOGY. HUMAN FACTORS CONSIDERATIONS THEN BECOME IMPORTANT BOTH FOR ECONOMIC AND PSYCHOLOGICAL REASONS. (A)

PRINCIPAL AREAS DISCUSSED ARE: "CONVERSATIONAL" LANGUAGES, SYSTEM RESPONSE TIME, CHARGING ALGORITHMS AND THEIR EFFECT ON SYSTEM USE, EASE OF USE AND CONFLICTING NEEDS OF NOVICE AND EXPERT USERS, AND MAXIMIZATION OF ACCESSIBILITY VERSUS MINIMIZATION OF SYSTEM IDLE TIME. (HRR)

7P, 4R.

292 SOFTWARE ENGINEERING

NOONAN, R.E. STRUCTURED PROGRAMMING AND FORMAL SPECIFICATION. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 421-424.

293 DOCUMENTATION

OKIMOTO, G.H., THE EFFECTIVENESS OF COMMENTS: A PILOT STUDY (SDD TECHNICAL REPORT NO. TR01.1347). ENDICOTT, NEW YORK: IBM CORP., SYSTEMS DEVELOPMENT DIVISION, JULY 1970.

DESCRIPTION:

THIS STUDY INVESTIGATES METHODS OF QUANTIFYING THE EFFECTIVENESS OF COMMENTS ACCOMPANYING IBM SYSTEM/360 ASSEMBLER LANGUAGE INSTRUCTIONS WHILE LIMITING THE SCOPE OF THE COMMENT PROBLEMS. SEVERAL EXPERIMENTS WERE PERFORMED WITH EXPERIENCED PROGRAMMERS WHICH TEND TO INDICATE THAT COMMENTS DO INDEED INFLUENCE PROGRAMMER PERFORMANCE; HOWEVER, NOT NECESSARILY ACCORDING TO BELIEFS. AN INVESTIGATION TO FURTHER SUBSTANTIATE THE RESULTS OF THIS PILOT STUDY IS CURRENTLY UNDERWAY. (A)

12P, 9R.

294 SOFTWARE DEVELOPMENT

ORTEGA, L.H. STRUCTURED PROGRAMMING SERIES (VOL. 7): DOCUMENTATION STANDARDS. (REPORT NO. RADC-TR-74-300-VOL-7). GRIFFISS AFB, NEW YORK: ROME AIR DEVELOPMENT CENTER, SEPTEMBER 1974. (NTIS NO. AD A008639)

DESCRIPTION:

THIS FINAL REPORT CONTAINS THE FULL STUDY FINDINGS FOR SOW TASK 4.1.7. INCLUDED ARE PROPOSED CHANGES TO DOD DOCUMENTATION STANDARDS NECESSARY TO REALIZE THE BENEFITS OF STRUCTURED PROGRAMMING TECHNOLOGY AS RELATED TO DOCUMENTATION. THE RECOMMENDED CHANGES TO USAF MIL-STD-483 AND DOD 4120.17M CONSTITUTE THE INITIAL STEP IN IMPROVING SOFTWARE DOCUMENTATIONS. (A)

100P, 11R.

295 SOFTWARE PHYSICS
OTTENSTEIN, L.M. FURTHER VALIDATION OF AN ERROR HYPOTHESIS. SOFTWARE ENGINEERING NOTES, JANUARY 1978, 3(1), 27-28.
DESCRIPTION:
SOFTWARE PHYSICS HYPOTHESES ARE USED TO PREDICT IMPLEMENTATION TIME AND NUMBER OF ERRORS FOR A PROGRAM MODULE. PREDICTIONS ARE COMPARED WITH THE OBSERVED VALUES REPORTED BY M.L. SHOOMAN AND M.I. BOLSKY (1975). (MEA)
2P, 7R.

296 SOFTWARE PHYSICS
OTTENSTEIN, L.M., SCHNEIDER, V.B., & HALSTEAD, M.H. PREDICTING THE NUMBER OF BUGS EXPECTED IN A PROGRAM MODULE (TECHNICAL REPORT NO. CSD-TR-205). WEST LAFAYETTE, INDIANA: PURDUE UNIVERSITY, JANUARY 1977.

297 PROGRAMMING, MAINTENANCE
OVERTON, R.K., ET. AL. DEVELOPMENT IN COMPUTER AIDED SOFTWARE MAINTENANCE (REPORT NO. ESD-TR-74-307). HANSCOM AFB, MASSACHUSETTS: DEPUTY FOR COMMAND AND MANAGEMENT SYSTEMS, HQ ELECTRONIC SYSTEMS DIVISION (AFSC), 1974.
DESCRIPTION:
DATA WERE COLLECTED ON TWO ASPECTS OF MAINTENANCE PROGRAMMING (WHICH, ACCORDING TO PUBLISHED ESTIMATES, COSTS THE U.S. APPROXIMATELY FIVE BILLION DOLLARS A YEAR). ASPECTS WERE (1) ARRANGEMENT AND SOURCES OF INFORMATION AT GRAPHICS CONSOLES, AND (2) THE VALUE OF "CONCEPTUAL GROUPINGS" TO MAINTENANCE PROGRAMMERS USING FORTRAN AND PL/1. (A)
263P, 60R.

298 SOFTWARE MAINTENANCE
OVERTON, R.K., COLEN, P., FREEMAN, P., WERSAN, S.J., VEIGEL, M.L., & STEELMAN, R. RESEARCH TOWARD WAYS OF IMPROVING SOFTWARE MAINTENANCE: RICASM FINAL REPORT (TECHNICAL REPORT NO. ESD-TR-73-125). CLAREMONT, CALIFORNIA: CORPORATION FOR INFORMATION SYSTEMS RESEARCH AND DEVELOPMENT, JANUARY 1973. (NTIS NO. AD 760819)
DESCRIPTION:
AS STEPS TOWARD MAKING IT EASIER TO MAINTAIN COMPUTER PROGRAMS, STUDIES WERE MADE OF SOME FUNDAMENTAL ASPECTS OF THE WORK. THE EVIDENCE INDICATES THAT (1) BEFORE A PERSON CAN MODIFY A PROGRAM EFFICIENTLY, HE NEEDS TO BE ABLE TO TRACE THE STRUCTURE INTO WHICH THE MODIFICATION HAS TO FIT, AND RECOGNIZE THE CONCEPTUAL BLOCKS OF WHICH THE STRUCTURE IS BUILT; (2) IT SHOULD BE POSSIBLE TO SPECIFY, AND SET SOME STANDARDS FOR, MAINTAINABILITY-AFFECTING FEATURES OF PROGRAMMING LANGUAGES, AND THE STYLE AND STRUCTURE OF PROGRAMS; (3) PHYSICAL CHARACTERISTICS OF TERMINAL DISPLAYS CAN HANDICAP OR HELP THE MAINTENANCE PROGRAMMER, AND DISPLAYS OF LISTS OF CUES AND POSSIBILITIES MAY ALSO HELP HIM. FUTURE DEVELOPMENTS, BASED ON THESE POINTS, WERE RECOMMENDED. (A)
183P, 47R.

299 PROGRAMMER PRODUCTIVITY
OYER, P.D. EVIDENCE OF INCREASED PROGRAMMER PRODUCTIVITY THROUGH USE OF KEYBOARD TERMINALS WITH DIRECT ACCESS TO COMPUTERS. UNPUBLISHED MANUSCRIPT, OCTOBER 1976 (AVAILABLE FROM U.S. BUREAU OF THE CENSUS, WASHINGTON, D.C.).

300 SOFTWARE DESIGN
PACKER, D.W. EFFECTIVE PROGRAM DESIGN. COMPUTERS AND PEOPLE, MARCH 1974, 23(3), PP. 3; 16-19; 41.
DESCRIPTION:
THIS DISCUSSION OF COMPUTER PROGRAM DESIGN PROMOTES THE IDEA THAT THE SUCCESS AND ULTIMATELY THE COST OF ANY COMPUTER PROGRAM IS CRITICALLY RELATED TO ITS DESIGN; THAT IS, THAT THE DESIGN IS, BY FAR, THE MOST CRITICAL ASPECT OF PROGRAM DEVELOPMENT. THE EDP COMMUNITY OFTEN TALKS OF DESIGN AND THE CONCEPTS OF MODULARITY, GENERALITY, FLEXIBILITY, AND MAINTAINABILITY; YET MANY PROGRAMS ARE NOT WELL DESIGNED AT ALL, BUT SIMPLY WRITTEN. IT IS MY BELIEF THAT THE DESIGN OF A PROGRAM IS MUCH DIFFERENT FROM ITS CODING, AND IS A CREATIVE TASK INVOLVING MANY -- IF NOT ALL -- OF THE SAME ELEMENTS AS SYSTEMS DESIGN. (A)
5P, OR.

301 PROGRAMMING LANGUAGES
PALME, J. PROGRAMMING LANGUAGES FOR THE SOFTWARE ENGINEER. IN INFOTECH INFORMATION LTD., SOFTWARE ENGINEERING. BERKSHIRE, ENGLAND: INFOTECH INFORMATION LTD., 1972, 463-494.

302 PROGRAMMING LANGUAGES
PALME, J. LANGUAGES FOR RELIABLE SOFTWARE. DATAMATION, DECEMBER 1975, 21(12), 77-80.
DESCRIPTION:
A NUMBER OF PROGRAMMING LANGUAGE AND OPERATING SYSTEM CONSTRUCTS ARE AVAILABLE WHICH MAY INCREASE THE RELIABILITY OF SOFTWARE, BUT WHICH ARE NOT WIDELY USED, EVEN AMONG THE MOST POPULAR LANGUAGES. THESE CONSTRUCTS ALLOW GREATER DETECTION OF ERRORS AT EXECUTION TIME, OR, IN SOME CASES, AT COMPILATION TIME. AMONG THE FEATURES DISCUSSED ARE RESTRICTIONS ON DATA TYPES AND VALUES, FEATURES WHICH PREVENT UNDEFINED CALCULATIONS, DATA TYPE AND RANGE CHECKING AT MODULE INTERFACES, AND AVOIDANCE OF SEVERAL CONSTRUCTS (E.G., INTERRUPTS, UNNECESSARY PARALLEL PROCESSING) WHICH HAVE BEEN OBSERVED TO CAUSE PROBLEMS. MANY OF THE RECOMMENDED FEATURES HAVE BEEN IMPLEMENTED IN THE LANGUAGE SIMULA 67. (HRR)
3P, OR.

303 SOFTWARE DEVELOPMENT
PARNAS, D. INFORMATION DISTRIBUTION ASPECTS OF DESIGN METHODOLOGY. PROCEEDINGS OF IFIP CONGRESS, 1971, 71, 339-344.
DESCRIPTION:
THE ROLE OF DOCUMENTATION IN THE DESIGN AND IMPLEMENTATION OF COMPLEX SYSTEMS IS EXPLORED, RESULTING IN SUGGESTIONS IN SHARP CONTRAST WITH CURRENT PRACTICE. THE CONCEPT OF SYSTEM STRUCTURE IS STUDIED BY EXAMINING THE MEANING OF THE PHRASE "CONNECTIONS BETWEEN MODULES". IT IS SHOWN THAT SEVERAL SYSTEM DESIGN GOALS (EACH SUGGESTING A PARTIAL TIME ORDERING OF THE DECISIONS) MAY BE INCONSISTENT. SOME PROPERTIES OF PROGRAMMERS ARE DISCUSSED. SYSTEM DOCUMENTATION WHICH MAKES ALL INFORMATION ACCESSIBLE TO ANYONE WORKING ON THE PROJECT IS DISCUSSED. THE THESIS THAT SUCH INFORMATION "BROADCASTING" IS HARMFUL, THAT IT IS HELPFUL IF MOST SYSTEM INFORMATION CAN BE HIDDEN FROM MOST PROGRAMMERS, IS SUPPORTED BY USE OF THE ABOVE MENTIONED CONSIDERATIONS AS WELL AS BY EXAMPLES. (A)
6P, 18R.

304 SOFTWARE ENGINEERING
PARNAS, D.L. SOME CONCLUSIONS FROM AN EXPERIMENT IN SOFTWARE ENGINEERING TECHNIQUES. AFIPS CONFERENCE PROCEEDINGS, 1972, 41, 325-329.
DESCRIPTION:
A SMALL PROGRAMMING PROJECT, UNDERTAKEN BY RELATIVELY INEXPERIENCED PROGRAMMERS, WAS DONE TO ASSESS SEVERAL PARAMETERS AND CLAIMS OF STRUCTURED PROGRAMMING TECHNIQUES. THE DESIGN INCLUDED FIVE MODULES TO BE PROGRAMMED BY EACH OF FIVE TECHNIQUES. A NUMBER OF THESE WERE INCORRECT OR INCOMPLETE. RESULTS WERE DRAWN FROM THE REST. THE PAPER DOES NOT EXPLAIN THE EXPERIMENTAL DESIGN; INSTEAD, IT DWELLS MAINLY ON THE CONCLUSIONS. THESE ARE THAT EFFORT SHOULD BE HEAVILY INVESTED IN THE PRE-DESIGN PHASE FOR THE GREATEST RETURNS. DOCUMENTATION OF EXTERNALS, AND THAT STEMMING FROM PRE-DESIGN, WAS THE MOST VALUABLE. MODULE TESTING BEFORE INTEGRATION SEEMED VALUABLE, AND IT SHOULD BE DONE BY OTHER THAN THE ORIGINAL PROGRAMMER. THE HYPOTHESIS THAT SINGLE INPUT/OUTPUT SUBROUTINE CALLS SHOULD BE THE ONLY COMMUNICATION BETWEEN MODULES WAS REJECTED. DATA STRUCTURES, HOWEVER, WERE KEPT WITHIN SINGLE MODULES. (GDC)
5P, 8R.

305 SOFTWARE DEVELOPMENT
PARNAS, D.L. A TECHNIQUE FOR SOFTWARE MODULE SPECIFICATION WITH EXAMPLES. COMMUNICATIONS OF THE ACM, 1972, 15, 330-336.
DESCRIPTION:
THIS PAPER PRESENTS AN APPROACH TO WRITING SPECIFICATIONS FOR PARTS OF SOFTWARE SYSTEMS. THE MAIN GOAL IS TO PROVIDE SPECIFICATIONS SUFFICIENTLY PRECISE AND COMPLETE THAT OTHER PIECES OF SOFTWARE CAN BE WRITTEN TO INTERACT WITH THE PIECE SPECIFIED WITHOUT ADDITIONAL INFORMATION. THE SECONDARY GOAL IS TO INCLUDE IN THE SPECIFICATION NO MORE INFORMATION THAN NECESSARY TO MEET THE FIRST GOAL. THE TECHNIQUE IS ILLUSTRATED BY MEANS OF A VARIETY OF EXAMPLES FROM A TUTORIAL SYSTEM. (A)
7P, 6R.

306 PROGRAMMING METHODOLOGY
PARNAS, D.L. ON THE DESIGN AND DEVELOPMENT OF PROGRAM FAMILIES. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1976, SE-2, 1-8.

307 SOFTWARE DEVELOPMENT
PARNAS, D., & DARRINGER, J. SODAS AND A METHODOLOGY FOR SYSTEM DESIGN. AFIPS CONFERENCE PROCEEDINGS, 1967, 31.

308 GENERAL DISCUSSION OF HUMAN FACTORS IN COMPUTER SYSTEMS
PARSONS, H.M. THE SCOPE OF HUMAN FACTORS IN COMPUTER-BASED DATA PROCESSING SYSTEMS. HUMAN FACTORS, 1970, 12, 165-175.
DESCRIPTION:
WORK IN HUMAN FACTORS ENCOMPASSES RESEARCH AND APPLICATION IN HUMAN ENGINEERING, PROCEDURE DEVELOPMENT, TRAINING TECHNIQUES, PERSONNEL REQUIREMENTS, TEST AND EVALUATION, TASK DESCRIPTION, AND TASK ALLOCATION. OPPORTUNITIES AND NEEDS EXIST IN COMPUTER-BASED DATA PROCESSING SYSTEMS FOR ALL THESE ENDEAVORS, ESPECIALLY WITH REGARD TO ON-LINE USERS. WITHIN HUMAN ENGINEERING, ONLY MANUAL ENTRY HAS SO FAR RECEIVED MUCH RESEARCH ATTENTION. WORK IS ALSO NEEDED ON DISPLAYS, INTEGRATED ENTRY-DISPLAY, WORKSPACE AND OTHER EQUIPMENT ASPECTS, ON-LINE LANGUAGES, AND PROGRAM PRODUCTION. OF GREATEST CONCERN TO HUMAN ENGINEERING IS THE COMPUTER OUTPUT, DESIGNED BY PROGRAMMERS, RATHER THAN THE HARDWARE. HUMAN FACTORS PEOPLE WILL HAVE TO MASTER A NEW FIELD AND PROVIDE GUIDANCE TO A NEW DISCIPLINE WHICH HAS NOT YET UNDERSTOOD HUMAN FACTORS REQUIREMENTS. (A)
11P, 7R.

309 DESIGN REVIEWS

PERRIENS, M.P. AN APPLICATION OF FORMAL INSPECTIONS TO TOP-DOWN STRUCTURED PROGRAM DEVELOPMENT (TECHNICAL REPORT RADC-TR-77-212). GAITHERSBURG, MARYLAND: IBM FEDERAL SYSTEMS DIVISION, JUNE 1977. (NTIS NO. AD AC41645)

DESCRIPTION:

THIS REPORT CONTAINS THE FULL STUDY FINDINGS FOR SOW TASK 4.3. THE RESULTS OF THE STUDY WERE THAT TOP-DOWN STRUCTURED PROGRAMMING AND FORMAL INSPECTIONS ARE COMPATIBLE AND CAN BE USED IN COMBINATION DURING SOFTWARE DEVELOPMENT. PRIOR TO RECOMMENDING FULL-SCALE USE, RADC SHOULD IMPLEMENT THE RECOMMENDED INSPECTION METHODOLOGY ON A VARIETY OF SELECTED SOFTWARE PROJECTS USING TOP-DOWN STRUCTURED PROGRAMMING. (A)

310 COMPUTER PROGRAMMER SELECTION

PERRY, D.K., & CANNON, W.M. A VOCATIONAL INTEREST SCALE FOR COMPUTER PROGRAMMERS: FINAL REPORT. IN PROCEEDINGS OF THE 4TH ANNUAL COMPUTER PERSONNEL RESEARCH CONFERENCE, ASSOCIATION FOR COMPUTING MACHINERY, 1966, 61-82.

311 VOCATIONAL INTERESTS OF PROGRAMMERS

PERRY, D.K., & CANNON, W.M. VOCATIONAL INTERESTS OF COMPUTER PROGRAMMERS. JOURNAL OF APPLIED PSYCHOLOGY, 1967, 51, 28-34.

DESCRIPTION:

THE REVISED SVIB WAS ADMINISTERED TO 1,378 COMPUTER PROGRAMMERS. PRIMARY ANALYSES WERE LIMITED TO 1,003 MALES WITH AT LEAST 2 YR. OF PROGRAMMING EXPERIENCE, WHOSE JOBS WERE PRIMARILY NONSUPERVISORY, AND WHO INDICATED SATISFACTION WITH PROGRAMMING. PROGRAMMERS DIFFER FROM OTHER PROFESSIONAL MEN PRIMARILY IN THEIR GREATER INTEREST IN PROBLEM SOLVING, MATHEMATICS, AND MECHANICAL PURSUITS, AND THEIR LESSER INTEREST IN PEOPLE. THEIR INTERESTS ARE MOST SIMILAR TO OPTOMETRISTS, CHEMISTS, ENGINEERS, PRODUCTION MANAGERS, MATHEMATICS-SCIENCE TEACHERS, AND SENIOR CPAS; BUT NONE OF THESE EXISTING KEYS ADEQUATELY REPRESENTS THE INTERESTS OF PROGRAMMERS. A PROGRAMMER KEY DEVELOPED ON HALF THE SAMPLE AND EVALUATED ON THE REMAINING HALF DISCRIMINATES WELL BETWEEN PROGRAMMERS AND MEN IN GENERAL. SATISFIED PROGRAMMERS SCORE SIGNIFICANTLY HIGHER ON THE KEY THAN DISSATISFIED PROGRAMMERS. (A)

7P, 10R.

312 SOFTWARE DESIGN

PETERS, L.J., & TRIPP, L.L. IS SOFTWARE DESIGN WICKED? DATAMATION, MAY 1976, 22(5), PP. 127; 131; 136.

DESCRIPTION:

A COMPARISON OF THE ATTRIBUTES AND PROBLEMS ASSOCIATED WITH SOFTWARE DESIGN AND THE CHARACTERISTICS OF "WICKED" PROBLEMS CLEARLY ILLUSTRATES THAT SOFTWARE DESIGN IS A WICKED PROBLEM. TOP-DOWN DESIGN, AN APPROACH THAT HAS RECEIVED CONSIDERABLE ATTENTION AND WIDESPREAD ACCEPTANCE, DOES NOT APPEAR TO BE BASED ON ASSUMPTIONS THAT ARE COMPATIBLE WITH SUCH PROBLEMS. ALTHOUGH TOP-DOWN DESIGN OFFERS DISTINCT ADVANTAGES OVER OTHER DESIGN METHODS, MORE ATTENTION MUST BE DIRECTED TOWARD DEVELOPING METHODS DIRECTED TOWARD WICKED PROBLEMS. (MFA)

3P, 3R.

313 SOFTWARE DESIGN METHODOLOGIES

PETERS, L.J., & TRIPP, L.L. COMPARING SOFTWARE DESIGN METHODOLOGIES. DATAMATION, NOVEMBER 1977, 23(11), 89-94.

DESCRIPTION:

SOFTWARE DESIGN HAS EVOLVED TO THE STAGE WHERE SEVERAL METHODOLOGIES HAVE BEEN PROPOSED. THIS PAPER BRIEFLY REVIEWS STRUCTURED DESIGN, THE JACKSON METHODOLOGY, LOGICAL CONSTRUCTION OF PROGRAMS, META STEPWISE REFINEMENT, AND HIGHER ORDER SOFTWARE. IT IS CONCLUDED THAT NO SINGLE METHOD EXISTS THAT WOULD BE USEFUL IN EVERY DESIGN PROBLEM, THAT EACH METHOD MAKES UNPROVABLE ASSUMPTIONS, THAT METHODS ASSIST ONLY IN SOLVING ROUTINE ASPECTS OF A DESIGN PROBLEM, AND THAT DESIGNERS MAY BE RELUCTANT TO USE THESE METHODS. (MEA) 6P, 13R.

314 NATURAL LANGUAGE PROGRAMMING

PETRICK, S.R. ON NATURAL LANGUAGE BASED COMPUTER SYSTEMS. IBM JOURNAL OF RESEARCH AND DEVELOPMENT, 1976, 20, 314-325.

DESCRIPTION:

SOME OF THE ARGUMENTS THAT HAVE BEEN GIVEN BOTH FOR AND AGAINST THE USE OF NATURAL LANGUAGES IN QUESTION-ANSWERING AND PROGRAMMING SYSTEMS ARE DISCUSSED. SEVERAL NATURAL LANGUAGE BASED COMPUTER SYSTEMS ARE CONSIDERED IN ASSESSING THE CURRENT LEVEL OF SYSTEM DEVELOPMENT. FINALLY, CERTAIN PERVASIVE DIFFICULTIES THAT HAVE ARISEN IN DEVELOPING NATURAL LANGUAGE BASED SYSTEMS ARE IDENTIFIED, AND THE APPROACH TAKEN TO OVERCOME THEM IN THE REQUEST (RESTRICTED ENGLISH QUESTION-ANSWERING) SYSTEM IS DESCRIBED. (A)

315 PERFORMANCE MODELS OF MAN-MACHINE SYSTEMS

PEW, R.W., BARON, S., FEEHRER, C.E., & MILLER, D.C. CRITICAL REVIEW AND ANALYSIS OF PERFORMANCE MODELS APPLICABLE TO MAN-MACHINE SYSTEMS EVALUATION (REPORT NO. 3446). CAMBRIDGE, MASSACHUSETTS: BOLT BERANEK AND NEWMAN, INC., MARCH 1977. (NTIS NO. AD A038597)

DESCRIPTION:

THIS REPORT FOCUSES ON THE REVIEW OF POTENTIALLY RELEVANT MODELS AND ON THE IDENTIFICATION OF ISSUES IN MODEL DEVELOPMENT AND APPLICATION THAT MAY HAVE IMPORTANT IMPACT ON MODELS FOR LARGE SCALE MAN-MACHINE SYSTEMS. A DETAILED AND CRITICAL EVALUATION OF SEVERAL CLASSES OF HUMAN PERFORMANCE MODELS IS PRESENTED. INTERRELATIONSHIPS AMONG EXISTING MODELS ARE EXAMINED AND AN EVALUATION IS MADE OF THE NEEDS AND GAPS IN THE TECHNOLOGY. MODELING ISSUES ARE IDENTIFIED AND RESEARCH RECOMMENDATIONS SUGGESTED. APPROXIMATELY FORTY MODELS OR MODEL TECHNIQUES THAT HAVE SOME APPLICABILITY TO THE SIMULATION MODELING PROGRAM ARE DESCRIBED IN THE APPENDIX. (A) 305P, 191R.

316 NATURAL LANGUAGE PROGRAMMING

PLATH, W.J. REQUEST: A NATURAL LANGUAGE QUESTION-ANSWERING SYSTEM. IBM JOURNAL OF RESEARCH AND DEVELOPMENT, 1976, 20, 326-335.

DESCRIPTION:

REQUEST IS AN EXPERIMENTAL RESTRICTED ENGLISH QUESTION-ANSWERING SYSTEM THAT CAN ANALYZE AND ANSWER A VARIETY OF ENGLISH QUESTIONS, SPANNING A SIGNIFICANT RANGE OF SYNTACTICAL COMPLEXITY, WITH RESPECT TO A SMALL FORTUNE 500 TYPE DATA BASE. THE LONG-RANGE OBJECTIVE OF THIS WORK IS TO EXPLORE THE POSSIBILITY OF PROVIDING NONPROGRAMMERS WITH A CONVENIENT AND POWERFUL MEANS OF ACCESSING INFORMATION IN FORMATTED DATA BASES WITHOUT HAVING TO LEARN A FORMAL QUERY LANGUAGE. TO ADDRESS THE SOMEWHAT CONFLICTING REQUIREMENTS OF UNDERSTANDABILITY FOR THE MACHINE AND MAXIMUM NATURALNESS FOR THE USER, REQUEST USES A LANGUAGE PROCESSING APPROACH FEATURING: 1) THE USE OF RESTRICTED ENGLISH; 2) A TWO-PHASE, COMPILER-LIKE ORGANIZATION; AND 3) LINGUISTIC ANALYSIS BASED ON A TRANSFORMATIONAL GRAMMAR. THE PRESENT PAPER EXPLORES THE MOTIVATION FOR THIS APPROACH IN SOME DETAIL AND ALSO DESCRIBES THE ORGANIZATION, OPERATION, AND CURRENT STATUS OF THE SYSTEM. (A)

317 SOFTWARE DESIGN
PLUM, T. A CASE STUDY COMPARISON OF TWO APPROACHES TO STRUCTURED DESIGN -- DATA-FLOW-MODULARITY VS. DATA-STRUCTURE-HIERARCHY. YOURDON REPORT, SEPTEMBER 1976, 1(7), 8-12.
DESCRIPTION:
AN EXPERIMENT WAS CONDUCTED TO COMPARE TWO POPULAR APPROACHES TO COMPUTER SYSTEM DESIGN -- DATA-FLOW-MODULARITY AND DATA-STRUCTURE-HIERARCHY. BOTH APPROACHES ARE PRODUCTIVE DESIGN TECHNIQUES, BUT EACH OFFERS UNIQUE ADVANTAGES. THE SYNTHESIS OF THESE TWO APPROACHES IS AN INTERESTING PROBLEM THAT COULD PRODUCE VERY PROMISING RESULTS. (MEA)
5P, 6R.

318 SOFTWARE DEVELOPMENT
POKORNEY, J.L., & MITCHELL, W.E. A SYSTEMS APPROACH TO COMPUTER PROGRAMS (TECHNICAL REPORT NO. 57-205). BEDFORD, MASSACHUSETTS: ELECTRONIC SYSTEMS DIVISION, TECHNICAL EQUIPMENTS AND STANDARDS OFFICE, L.G. HANSCOM FIELD, FEBRUARY 1967.

319 PROGRAMMING LANGUAGES
RALSTON, A.W., & WAGENER, J.L. STRUCTURED FORTRAN: AN EVOLUTION OF STANDARD FORTRAN. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1976, SE-2, 154-175.

320 PROGRAMMING
RAMAMOORTHY, C.V. IMPROVING THE EFFECTIVENESS OF COMMERCIAL PROGRAMMING (INFOTECH REPORT #19). 1972.

321 SOFTWARE ENGINEERING
RAMAMOORTHY, C.V., & HO, S.-B.F. TESTING LARGE SOFTWARE WITH AUTOMATED SOFTWARE EVALUATION SYSTEMS. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 46-58.
DESCRIPTION:
IN THE PAST FEW YEARS, RESEARCH HAS BEEN ACTIVELY CARRIED OUT IN AN ATTEMPT TO IMPROVE THE QUALITY AND RELIABILITY OF LARGE-SCALE SOFTWARE SYSTEMS. ALTHOUGH PROGRESS HAS BEEN MADE ON THE FORMAL PROOF OF PROGRAM CORRECTNESS, PROVING LARGE-SCALE SOFTWARE SYSTEMS CORRECT BY FORMAL PROOF IS STILL MANY YEARS AWAY. AUTOMATED SOFTWARE TOOLS HAVE BEEN FOUND TO BE VALUABLE IN IMPROVING SOFTWARE RELIABILITY AND ATTACKING THE HIGH COST OF SOFTWARE SYSTEMS. THIS PAPER ATTEMPTS TO DESCRIBE SOME MAIN FEATURES OF AUTOMATED SOFTWARE TOOLS AND SOME SOFTWARE EVALUATION SYSTEMS THAT ARE CURRENTLY AVAILABLE. (A)
13P, 31R.

322 SOFTWARE EVALUATION

RAMAMOORTHY, C.V., MEEKER, R.E., JR., & TURNER, J. DESIGN AND CONSTRUCTION OF AN AUTOMATED SOFTWARE EVALUATION SYSTEM. IEEE SYMPOSIUM ON COMPUTER SOFTWARE RELIABILITY, 1973, 28-37.

DESCRIPTION:

THE PROBLEM OF EVALUATING AND VALIDATING LARGE PROGRAMS AND PROGRAMMING SYSTEMS IS ONE THAT LENDS ITSELF TO SOME FORM OF AUTOMATIC ANALYSIS. THE CONCEPT OF AN AUTOMATED EVALUATION SYSTEM IS DEVELOPED AND ILLUSTRATED WITH THE DETAILED DESCRIPTION OF A CURRENT WORKING SYSTEM. THIS SYSTEM IS DESIGNED FOR AUTOMATIC ANALYSIS OF LARGE PROGRAMS AND INCLUDES BOTH STATIC AND DYNAMIC ANALYSIS. THE STATIC ANALYSIS CONSISTS OF A COMPLETE SCAN OF THE SOURCE CODE STATEMENTS OF A PROGRAM WITH AUTOMATIC RECOGNITION OF PREVIOUSLY DEFINED 'DANGEROUS CONDITIONS' AND THE CONSTRUCTION OF A DATA BASE OF PROGRAM CHARACTERISTICS. A GRAPH MODEL OF PROGRAM STRUCTURE IS CONSTRUCTED AND ANALYZED FOR WELL-FORMATION AND THE EXISTENCE OF LOOPS. DYNAMIC ANALYSIS CONSISTS OF THE AUTOMATIC INSERTION OF MONITORS FOR PRESCRIBED VARIABLES. A MONITORING SUBROUTINE CHECKS NEW VARIABLE VALUES WITH PRESCRIBED VALUES AND ALLOWS OTHER STATISTICS GATHERING FUNCTIONS AT RUN TIME. THE UTILITY OF THIS TYPE OF SYSTEM IN AN OVERALL EVALUATION EFFORT IS CONSIDERED ALONG WITH POSSIBILITIES FOR EXTENSIONS. (A)

10P, 3R.

323 PROGRAMMING LANGUAGES

RAMSEY, H.R. PLANS: HUMAN FACTORS IN THE DESIGN OF A COMPUTER PROGRAMMING LANGUAGE. IN PROCEEDINGS OF THE HUMAN FACTORS SOCIETY 18TH ANNUAL MEETING. SANTA MONICA, CALIFORNIA: HUMAN FACTORS SOCIETY, 1974, 39-41.

DESCRIPTION:

CONVENTIONAL COMPUTER PROGRAMMING LANGUAGES HAVE PROVEN INADEQUATE FOR USE IN THE SOLUTION OF COMPLEX RESOURCE ALLOCATION AND SCHEDULING PROBLEMS. A NEW PROGRAMMING LANGUAGE, PLANS, IS DESCRIBED. THE EMPHASIS IN THE DESIGN OF PLANS IS NOT ON SPECIALIZED SCHEDULING COMMANDS, BUT ON PROVISION OF APPROPRIATE BASIC DATA STRUCTURES FOR SCHEDULING PROBLEMS. A METHOD OF LANGUAGE SPECIFICATION IS DESCRIBED WHICH PROVIDES RIGOROUS FUNCTIONAL DEFINITION WHILE ALLOWING AN EXTREMELY CLEAN INTERFACE BETWEEN THE LANGUAGE FUNCTIONAL DESIGN PROCESS (IN WHICH HUMAN FACTORS PERSONNEL MIGHT PARTICIPATE MOST ACTIVELY) AND THE IMPLEMENTATION PROCESS. (A)

3P, 4R.

324 COMPARISON OF FLOWCHARTS AND PROGRAM DESIGN LANGUAGES

RAMSEY, H.R., ATWOOD, M.E., & VAN DOREN, J.R. A COMPARATIVE STUDY OF FLOWCHARTS AND PROGRAM DESIGN LANGUAGES FOR THE DETAILED PROCEDURAL SPECIFICATION OF COMPUTER PROGRAMS (TECHNICAL REPORT TR-78-A22). ARLINGTON, VIRGINIA: U.S. ARMY RESEARCH INSTITUTE FOR THE BEHAVIORAL AND SOCIAL SCIENCES, 1978.

DESCRIPTION:

AN EXPERIMENT WAS PERFORMED TO ASSESS THE RELATIVE MERITS OF PROGRAM DESIGN LANGUAGES (PDL'S) AND FLOWCHARTS AS TECHNIQUES FOR THE DEVELOPMENT AND DOCUMENTATION OF DETAILED DESIGNS FOR COMPUTER PROGRAMS.

TWENTY STUDENTS IN A COMPUTER SCIENCE GRADUATE COURSE PARTICIPATED IN THIS EXPERIMENT. WORKING INDIVIDUALLY, THE STUDENTS DESIGNED A TWO-PASS ASSEMBLER FOR A SIMPLE MINICOMPUTER. HALF OF THE STUDENTS EXPRESSED THEIR DESIGN FOR THE FIRST PASS OF THE ASSEMBLER IN THE FORM OF A FLOWCHART, AND EXPRESSED THEIR DESIGN FOR THE SECOND PASS IN A PROGRAM DESIGN LANGUAGE. THE OTHER HALF OF THE STUDENTS USED A PDL FOR PASS ONE, AND A FLOWCHART FOR PASS TWO. FLOWCHARTS AND PDL'S WERE COMPARED ON THE BASIS OF VARIOUS MEASURES OF OVERALL DESIGN QUALITY, DESIGN ERRORS, LEVEL OF DETAIL OF DESIGNS, TIME EXPENDED IN DEVELOPING DESIGNS, AND SUBJECTIVE PREFERENCES.

HAVING COMPLETED THIS DESIGN TASK, THE SUBJECTS THEN PERFORMED AN IMPLEMENTATION TASK. THEY WERE GIVEN FAIRLY DETAILED PROCEDURAL DESIGNS FOR A PROGRAM WHICH SIMULATES THE FUNCTION OF A FAIRLY SOPHISTICATED MINICOMPUTER. THEY WERE THEN REQUIRED TO DEVELOP A WORKING VERSION OF THE PROGRAM IN PL/1. ALTHOUGH THE DESIGNS WERE LOGICALLY EQUIVALENT, HALF THE STUDENTS RECEIVED THEIR SIMULATOR DESIGN IN FLOWCHART FORM, AND HALF IN PDL FORM. FLOWCHARTS AND PDL'S WERE COMPARED ON THE BASIS OF DESIGN COMPREHENSION TEST PERFORMANCE, VARIOUS MEASURES OF OVERALL IMPLEMENTATION QUALITY, IMPLEMENTATION ERRORS, AND SUBJECTIVE PREFERENCES.

IN THE CONTEXT IN WHICH THIS STUDY WAS PERFORMED, THE USE OF A PROGRAM DESIGN LANGUAGE (PDL) BY A SOFTWARE DESIGNER, FOR THE DEVELOPMENT AND DESCRIPTION OF A DETAILED PROGRAM DESIGN, PRODUCED BETTER RESULTS THAN DID THE USE OF FLOWCHARTS. SPECIFICALLY, THE DESIGNS APPEARED TO BE OF SIGNIFICANTLY BETTER QUALITY, INVOLVING MORE ALGORITHMIC OR PROCEDURAL DETAIL, THAN THOSE PRODUCED USING FLOWCHARTS. IN ADDITION, FLOWCHART DESIGNS EXHIBITED CONSIDERABLY MORE ABBREVIATION AND OTHER SPACE-SAVING PRACTICES THAN DID PDL DESIGNS, WITH A POSSIBLE ADVERSE EFFECT ON THEIR READABILITY.

WHEN EQUIVALENT, HIGHLY READABLE DESIGNS WERE PRESENTED TO SUBJECTS IN BOTH PDL AND FLOWCHART FORM, NO PATTERN OF SHORT-TERM OR LONG-TERM DIFFERENCES IN COMPREHENSION OF THE DESIGN WAS OBSERVED. NO SIGNIFICANT DIFFERENCES WERE DETECTED IN THE QUALITY OR OTHER PROPERTIES OF PROGRAMS WRITTEN AS IMPLEMENTATIONS OF THE DESIGNS. SUBJECTIVE RATINGS INDICATED A MILD PREFERENCE FOR PDL'S.

OVERALL, THE RESULTS SUGGEST THAT SOFTWARE DESIGN PERFORMANCE AND DESIGNER-PROGRAMMER COMMUNICATION MIGHT BE SIGNIFICANTLY IMPROVED BY THE ADOPTION OF INFORMAL PROGRAM DESIGN LANGUAGES, RATHER THAN FLOWCHARTS, AS A STANDARD DOCUMENTATION METHOD FOR DETAILED COMPUTER PROGRAM DESIGNS. (A)

325 INTERACTIVE PROGRAMMING

REASER, J., PRIESMAN, I., & GILL, J. A PRODUCTION ENVIRONMENT EVALUATION OF INTERACTIVE PROGRAMMING (TECHNICAL REPORT NO. USACSC-AT-74-C3). FORT BELVOIR, VIRGINIA: U.S. ARMY COMPUTER SYSTEMS COMMAND, DECEMBER 1974.

DESCRIPTION:

THIS REPORT DESCRIBES AN EVALUATION OF INTERACTIVE PROGRAMMING VERSUS BATCH PROGRAMMING WITHIN AN ACTUAL SOFTWARE PRODUCTION ENVIRONMENT AT THE U.S. ARMY COMPUTER SYSTEMS COMMAND. THE PURPOSE OF THE STUDY WAS TO DETERMINE THE PRODUCTIVITY AND COST EFFECTIVENESS DIFFERENCES BETWEEN THE TWO MODES OF OPERATION. THE STUDY WAS CONDUCTED IN A PRODUCTION ENVIRONMENT WITH THE PROGRAMMERS COMPLETING THEIR NORMALLY REQUIRED WORKLOAD AND PROVIDING DOCUMENTATION ON TASKS ROUTINELY ASSIGNED TO THEM. THE RESULTS OF THIS STUDY INDICATE THAT ON A LINE-OF-CODE BASIS THE INTERACTIVE SYSTEM OFFERS AN INCREASE IN PRODUCTIVITY AND A DECREASE IN OVERALL COST, AS COMPARED TO THE BATCH PROCESSING PROCEDURE. CONTINUED SUPPORT OF THE INTERACTIVE SYSTEM IS RECOMMENDED. (A)

71P, 7R.

326 SOFTWARE ENGINEERING

REIFER, D.J. AUTOMATED AIDS FOR RELIABLE SOFTWARE. IN PROCEEDINGS, 1975 INTERNATIONAL CONFERENCE ON RELIABLE SOFTWARE. SIGPLAN NOTICES, JUNE 1975, 10(6), 131-142.

DESCRIPTION:

RECENT INVESTIGATIONS ON THE USE OF AUTOMATION TO REALIZE THE TWIN OBJECTIVES OF COST REDUCTION AND RELIABILITY IMPROVEMENT FOR COMPUTER PROGRAMS DEVELOPED FOR THE U.S. AIR FORCE ARE REPORTED. THE CONCEPTS OF RELIABILITY AND AUTOMATION AS THEY PERTAIN TO SOFTWARE ARE EXPLAINED. THEN, OVER TWENTY AUTOMATED TOOLS AND TECHNIQUES (AIDS) IDENTIFIED BY THIS INVESTIGATION ARE DESCRIBED AND CATEGORIZED. BASED ON THE INFORMATION REVIEWED, AN ASSESSMENT OF THE STATE OF THE TECHNOLOGY IS MADE. FINALLY, SPECIFIC RECOMMENDATIONS WHICH TRY TO GIVE DIRECTION TO FUTURE EFFORTS ARE OFFERED. (A)

12P, 46R.

327 PROGRAMMER SELECTION

REINSTEDT, R.N., HAMMIDI, B.C., PERES, S.H., & RICARD, E.L. COMPUTER PERSONNEL RESEARCH GROUP PROGRAMMER PERFORMANCE PREDICTION STUDY (REPORT NO. RM-4033-PR). SANTA MONICA, CALIFORNIA: THE RAND CORP., MARCH 1964 (ALSO REPORTED MORE BRIEFLY IN REINSTEDT, R.N. RESULTS OF A PROGRAMMER PERFORMANCE PREDICTION STUDY. IEEE TRANSACTIONS ON ENGINEERING MANAGEMENT, DECEMBER 1967, EM-14, 183-187).

DESCRIPTION:

THIS MEMORANDUM REPORTS THE RESULTS OF A RESEARCH STUDY UNDERTAKEN TO GAIN SOME INSIGHT INTO THE RELATIONSHIP BETWEEN, ON THE ONE HAND, RATED JOB PERFORMANCE AND, ON THE OTHER, COGNITIVE ABILITIES, VOCATIONAL INTERESTS, AND BIOGRAPHICAL INFORMATION OF COMPUTER PROGRAMMERS.

A TEST BATTERY COMPOSED OF THE IBM PROGRAMMER APTITUDE TEST, THE TEST OF SEQUENTIAL INSTRUCTIONS (A RESEARCH INSTRUMENT SPECIALLY CONSTRUCTED FOR THIS STUDY), THE STRONG VOCATIONAL INTEREST BLANK, AND A PERSONAL BACKGROUND DATA FORM WAS ADMINISTERED TO 534 PROGRAMMERS REPRESENTING 24 PARTICIPATING COMPANIES. THE SAMPLE WAS DIVIDED INTO TWO SUB-SAMPLES: 301 PROGRAMMERS WERE CLASSIFIED AS SCIENTIFIC PROGRAMMERS AND 233 WERE CLASSIFIED AS BUSINESS PROGRAMMERS.

THE ANALYSES OF THE DATA INDICATE THAT THE IBM PROGRAMMER APTITUDE TEST AND THE TEST OF SEQUENTIAL INSTRUCTIONS GENERALLY CORRELATE HIGHER TO THE PERFORMANCE RATING CRITERION AMONG THE COMPANIES IN THE SCIENTIFIC SAMPLE THAN IN THE BUSINESS SAMPLE. WHEN THE DATA ARE ANALYZED AGAINST INDIVIDUAL COMPANY PERFORMANCE RATINGS, THERE IS A WIDE RANGE OF CORRELATION (-.67 TO .90).

THE ANALYSES OF THE STRONG VOCATIONAL INTEREST BLANK INDICATE THAT THIS INSTRUMENT HAS PREDICTIVE VALUE IN TERMS OF JOB PERFORMANCE IN BOTH SUB-SAMPLES; FURTHER, THE EMERGING INTEREST PATTERNS TEND TO BE THOSE OF THE SCIENTIFIC, PROFESSIONALLY-ORIENTED, ESTHETIC PERSON. BOTH SAMPLES SHOWED AN EXTREMELY HIGH INTEREST IN MATHEMATICAL SUBJECTS. HIGHEST CORRELATIONS RESULTED FROM THE BUSINESS PROGRAMMERS WHO HAD HIGHLY SCIENTIFIC INTEREST PATTERNS.

SCIENTIFIC PROGRAMMERS HAD A GENERALLY HIGHER EDUCATIONAL LEVEL THAN THE BUSINESS PROGRAMMERS (MORE COLLEGE GRADUATES AND MORE ADVANCED DEGREES).

PROGRAMMERS FROM BOTH THE SCIENTIFIC AND THE BUSINESS SAMPLES WERE, ON THE AVERAGE, A COMPARATIVELY YOUNG GROUP, WITH (AS COMPARED WITH OTHER PROFESSIONS) FEW YEARS OF EXPERIENCE.

FOLLOW-UP STUDIES ARE INDICATED, ESPECIALLY IN THE AREAS OF INTEREST AND PERSONALITY TESTING. (A)

64P, 4R.

328 QUERY LANGUAGES

REISNER, P. USE OF PSYCHOLOGICAL EXPERIMENTATION AS AN AID TO DEVELOPMENT OF A QUERY LANGUAGE. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, MAY 1977, SE-3, 218-229.

DESCRIPTION:

THIS PAPER DESCRIBES A SERIES OF PSYCHOLOGICAL EXPERIMENTS USED TO TEST A NEW DATA BASE QUERY LANGUAGE. THE INTENT IS TO MAKE PSYCHOLOGICAL TESTING OF A LANGUAGE PART OF THE DESIGN AND DEVELOPMENT PROCESS. BY TESTING A LANGUAGE WHILE IT IS STILL UNDER DEVELOPMENT, FEATURES THAT REQUIRE CHANGING CAN BE IDENTIFIED AND THE CHANGES MADE.

THE EXPERIMENTS, WHICH USED COLLEGE STUDENTS AS SUBJECTS, INVESTIGATED: (1) OVERALL "LEARNABILITY" OF THE LANGUAGE, (2) "LEARNABILITY" OF INDIVIDUAL FEATURES OF THE LANGUAGE, AND (3) THE TYPES AND FREQUENCIES OF ERRORS MADE. RECOMMENDED CHANGES TO THE PARTICULAR LANGUAGE AND THE BASIS FOR THOSE RECOMMENDATIONS ARE DESCRIBED.

IN ADDITION, ERRORS MADE BY EXPERIMENTAL SUBJECTS DURING THE TESTING ARE THEN ANALYZED FROM THE POINT OF VIEW OF POSSIBLE CAUSES. BASED ON THIS ANALYSIS, A PRELIMINARY MODEL OF QUERY WRITING AND TWO POSSIBLE INDICES OF QUERY COMPLEXITY ARE SUGGESTED. (A)

12P, 22R.

329 QUERY LANGUAGES

REISNER, P., BOYCE, R.F., & CHAMBERLIN, D.D. HUMAN FACTORS EVALUATION OF TWO DATA BASE QUERY LANGUAGES: SQUARE AND SEQUEL. AFIPS CONFERENCE PROCEEDINGS, 1975, 44, 447-452.

DESCRIPTION:

A SERIES OF EXPERIMENTS WAS CONDUCTED TO EVALUATE THE LEARNABILITY OF TWO DATA BASE QUERY LANGUAGES, SQUARE AND SEQUEL, USING UNIVERSITY STUDENTS AS SUBJECTS. SUBJECTS WITH OR WITHOUT PROGRAMMING EXPERIENCE WERE ABLE TO USE EITHER PROGRAMMING LANGUAGE WITH REASONABLE PROFICIENCY AFTER 12 OR 14 ACADEMIC HOURS OF INSTRUCTION. PROGRAMMERS LEARNED BOTH LANGUAGES MORE QUICKLY AND MORE COMPLETELY THAN DID NON-PROGRAMMERS, AND THE NON-PROGRAMMERS SHOWED GREATER PROFICIENCY WITH SEQUEL THAN WITH SQUARE. BOTH PROGRAMMERS AND NON-PROGRAMMERS WERE ABLE TO COMBINE BASIC LANGUAGE FEATURES IN WAYS NOT EXPLICITLY TAUGHT.

THE BASIC LANGUAGE FEATURE, A SIMPLE MAPPING, WAS LEARNED IN EACH LANGUAGE WITH NEAR-PERFECT ACCURACY BY PROGRAMMERS AFTER TWO HOURS AND BY NON-PROGRAMMERS AFTER FOUR HOURS. HOWEVER, CONSIDERABLE DIFFICULTY WAS EXPERIENCED IN LEARNING AND RETAINING MORE COMPLEX FEATURES, ESPECIALLY BY NON-PROGRAMMERS. A STUDY OF ERRORS MADE BY SUBJECTS SUGGESTS THAT A REAL DATA BASE SYSTEM SHOULD BE PREPARED TO CORRECT MINOR SYNTACTIC ERRORS AND TO SEARCH FOR POORLY-SPECIFIED DATA VALUES BY SOME TECHNIQUE SUCH AS STEM-MATCHING OR A SYNONYM DICTIONARY. (A, ABBR.)

6P, 3R.

330 PROGRAMMING LANGUAGES

RICHARD, F., & LEDGARD, H.F. A REMINDER FOR LANGUAGE DESIGNERS. SIGPLAN NOTICES, DECEMBER 1977, 12(12), 73-82 (ALSO COINS TECHNICAL REPORT NO. 76-3, AMHERST, MASSACHUSETTS: UNIVERSITY OF MASSACHUSETTS, DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION, MARCH 1976).

DESCRIPTION:

MOST PROGRAMMING LANGUAGES DO NOT SUIT THE PRODUCTION OF LARGE SOFTWARE SYSTEMS. TO IMPLEMENT REAL PROBLEMS, NO CURRENT PROGRAMMING LANGUAGE OFFERS CLEAR SOLUTIONS. TOO OFTEN, THE STRUCTURE OF THE PROBLEM MUST BE TWISTED TO THE STRUCTURE OF THE LANGUAGE. TO DEVISE PROGRAMS THAT CAN BE READ AND MODIFIED, PROGRAMMERS MUST OBSERVE STRICT OBEDIENCE TO NUMEROUS AND COMPLEX PROGRAMMING STANDARDS.

WE BELIEVE THERE IS A NEED FOR A NEW GENERAL PURPOSE, PROCEDURE-ORIENTED PROGRAMMING LANGUAGE -- "UTOPIA 84". IN THIS PAPER, WE SUGGEST SEVERAL LANGUAGE DESIGN PRINCIPLES FOR UTOPIA 84. (A, ABBR.)

10P, 19R.

331 SOFTWARE METRICS
RICHARDS, P. INTERIM REPORT ON A SOFTWARE COMPLEXITY MEASURE (SOFTWARE ERROR CONTROL IR&D PROJECT NO. 76D6A51). NOVEMBER 1976.

332 COMPUTER PERSONNEL SELECTION
RIGNEY, J.W., BERGER, R.M., & GERSHON, A. COMPUTER PERSONNEL SELECTION AND CRITERION DEVELOPMENT: I. THE RESEARCH PLANS (TECHNICAL REPORT 36). LOS ANGELES, CALIFORNIA: UNIVERSITY OF SOUTHERN CALIFORNIA, FEBRUARY 1963.

333 COMPUTER PERSONNEL SELECTION
RIGNEY,J.W., BERGER, R.M., GERSHON, A., & WILSON, R.C. COMPUTER PERSONNEL SELECTION AND CRITERION DEVELOPMENT: II. DESCRIPTION AND CLASSIFICATION OF COMPUTER PROGRAMMER AND ANALYST JOBS (TECHNICAL REPORT 37). LOS ANGELES, CALIFORNIA: UNIVERSITY OF SOUTHERN CALIFORNIA, DECEMBER 1963.

334 COMPUTER PERSONNEL
RIGNEY, J.W., BERGER, R.M., WILSON, R.C., & TEPLITZKY, F. COMPUTER PERSONNEL SELECTION AND CRITERION DEVELOPMENT: III -- THE BASIC PROGRAMMING KNOWLEDGE TEST (TECHNICAL REPORT NO. 49). LOS ANGELES, CALIFORNIA: UNIVERSITY OF SOUTHERN CALIFORNIA, ELECTRONICS PERSONNEL RESEARCH GROUP, JUNE 1966.
(NTIS NO. AD 636988)
DESCRIPTION:
THIS IS A REPORT ON THE CRITERION DEVELOPMENT PHASE OF A LONG-TERM RESEARCH PROGRAM CONCERNED WITH COMPUTER PERSONNEL SELECTION AND EVALUATION.
TWO TYPES OF CRITERION MEASURES ARE INVOLVED IN THIS PHASE. BOTH ARE PROFICIENCY TESTS, ONE DESIGNED TO TEST AN INDIVIDUAL'S KNOWLEDGE OF THE BASIC PRINCIPLES AND TECHNIQUES OF PROGRAMMING, AND THE SECOND, TO TEST AN INDIVIDUAL'S PERFORMANCE IN DEPTH IN THE SYSTEMS ANALYSIS AND SYSTEMS DESIGN AREAS. THE DEVELOPMENT OF THE FIRST TYPE, THE BASIC PROGRAMMING KNOWLEDGE TEST (BPKT), IS DESCRIBED IN THIS REPORT. THE SECOND TYPE OF MEASURE IS NOW UNDER CONSTRUCTION AND WILL BE DESCRIBED IN A SUBSEQUENT REPORT.
THE BPKT IS INTENDED TO STAND BY ITSELF AS A CRITERION OF PROGRAMMING PROFICIENCY. TO ACHIEVE A CLOSE CORRESPONDENCE OF TEST CONTENT TO PROGRAMMING JOB REQUIREMENTS, SUBJECT-MATTER EXPERTS PARTICIPATED IN THE CONSTRUCTION AND REVIEW OF THE TEST QUESTIONS. TEST QUESTIONS WERE SELECTED THAT MET THE CRITERIA OF DISCRIMINATION AND APPROPRIATE DIFFICULTY, AS INDICATED BY THE STATISTICAL ANALYSIS OF RESULTS OF A LARGE PRELIMINARY TESTING. THE FINAL FORM OF THE TEST CONSISTS OF 100 MULTIPLE-CHOICE QUESTIONS THAT ARE DESIGNED TO BE FREE OF REFERENCES TO SPECIFIC COMPUTERS AND LANGUAGES NOW IN USE.
NORMATIVE SCORES HAVE BEEN DEVELOPED FOR NAVY COMPUTER GROUPS. THE RELATIONSHIPS OF THE BPKT TEST SCORES TO A NUMBER OF VOCATIONAL AND EDUCATIONAL VARIABLES ARE DESCRIBED. RECOMMENDATIONS ARE MADE FOR THE USE OF THE BPKT IN PERSONNEL SELECTION AND EVALUATION OF EXPERIENCED PROGRAMMERS AND ANALYSTS, AND AS A CRITERION MEASURE AGAINST WHICH APTITUDE TESTS MAY BE VALIDATED. (A)
42P. 4R.

335 PROGRAM READABILITY
ROBERTS, K.V. PROGRAM READABILITY. IN INFOTECH INFORMATION LTD., SOFTWARE ENGINEERING. BERKSHIRE, ENGLAND: INFOTECH INFORMATION LTD., 1972, 495-516.

336 SOFTWARE DESIGN
ROBINSON, L. A FORMAL DESIGN MEDIUM FOR SOFTWARE. COMPUTER MAGAZINE, JUNE 1975, 8(6), 66.

337 PROGRAMMING LANGUAGES
ROBINSON, S.K., & TORSUN, I.S. AN EMPIRICAL ANALYSIS OF FORTRAN PROGRAMS. COMPUTER JOURNAL, 1976, 19, 56-62.
DESCRIPTION:
THIS PAPER DESCRIBES THE RESULTS OBTAINED FROM A STATIC ANALYSIS OF FORTRAN PROGRAMS WRITTEN AND RUN IN A UNIVERSITY ENVIRONMENT. THE ANALYSIS WAS PERFORMED BY A SYNTAX ANALYSER DESIGNED SPECIFICALLY TO ANALYSE SOURCE PROGRAM STATEMENTS, COLLECT DETAILED INFORMATION AND PRODUCE A REPORT CONCERNING THE TOTALS OF EACH FACTOR CONSIDERED.
THE AIM OF THE AUTHORS' RESEARCH IS TO PRODUCE AN OPTIMISING COMPILER DESIGNED TO MEET THE FORTRAN PROGRAMMER AS HE IS, NOT AS HE IS EXPECTED TO BE. (A)
7P, 3R.

338 PROGRAMMING
ROBINSON, S.K., & TORSUN, I.S. THE AUTOMATIC MEASUREMENT OF THE RELATIVE MERITS OF STUDENT PROGRAMS. SIGPLAN NOTICES, APRIL 1977, 12(4), 80-93.

339 COMPUTER PERSONNEL
ROEMMICH, H. TESTING PROGRAMMER EFFICIENCY. JOURNAL OF DATA MANAGEMENT, DECEMBER 1963, 1, 24-26.
DESCRIPTION:
ON NINE TESTS THAT EMPLOYERS THOUGHT WOULD MEASURE FACTORS RELEVANT TO COMPUTER PROGRAMMER PROFICIENCY, FOUR HAVE BEEN FOUND SIGNIFICANT BY DIFFERENTIATING COMPUTER PROGRAMMERS FROM OTHER "TYPES" OF PERSONS BY MEASURING A FACTOR THAT MIGHT BE CALLED "NUMERICAL INTELLIGENCE." THE TEST SCORES DO NOT TEND TO CORRELATE WITH SUPERVISOR RATINGS OF PROGRAMMER PROFICIENCY, HOWEVER. (A)
3P, 2R.

340 STRUCTURED PROGRAMMING
ROMANOS, J.P. AN IMPLEMENTATION OF STRUCTURED CODE TECHNIQUES ON A REAL-TIME SYSTEM. COMPUTER MAGAZINE, JUNE 1975, 8(6), 48-49.
DESCRIPTION:
THIS PAPER DESCRIBES EXPERIENCES USING STRUCTURED PROGRAMMING AND A MODIFIED TEAM CONCEPT. RESPONSIBILITIES WITHIN A TEAM WERE ROTATED SO THAT NO TEAM MEMBER CODED A MODULE HE HAD DESIGNED OR TESTED A MODULE THAT HE HAD DESIGNED OR CODED. ALTHOUGH CORE USAGE AND EXECUTION TIMES INCREASED SLIGHTLY, CODE WAS FOUND TO BE MORE READABLE AND UNDERSTANDABLE AND THE DEBUGGING PHASES OF A PROJECT WERE SIGNIFICANTLY AIDED. (MEA)
2P, 0R.

341 PROGRAMMING LANGUAGES
ROSEN, S. (ED) PROGRAMMING SYSTEMS AND LANGUAGES. NEW YORK: MCGRAW-HILL, 1967.

342 AUTOMATIC PROGRAMMING

ROSENSCHEIN, S.J., & KATZ, S.M. SELECTION OF REPRESENTATIONS FOR DATA STRUCTURES. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 147-154 (ALSO SIGART NEWSLETTER, AUGUST 1977, NO. 64, 147-154).

DESCRIPTION:

THE PROCESS OF SELECTING REPRESENTATIONS FOR DATA STRUCTURES IS CONSIDERED. THE MODEL OF THE SELECTION PROCESS WE SUGGEST IS CENTERED AROUND A BASE OF KNOWN ABSTRACT DATA STRUCTURES AND THEIR REPRESENTATIONS. THE ABSTRACT DATA STRUCTURE FOR WHICH A REPRESENTATION IS REQUIRED WOULD NOT NECESSARILY BE IN THE BASE, BUT SHOULD BE A COMBINATION OF BASE DATA STRUCTURES.

AFTER DESCRIBING THIS MODEL OF SELECTION AND ITS MOTIVATION, TWO ASPECTS OF THE PROCESS ARE EXAMINED IN MORE DETAIL: A) THE INTERACTION WITH THE USER IS TREATED BY DEFINING A LANGUAGE FOR THE NATURAL DESCRIPTION OF DATA STRUCTURE REQUIREMENTS, AND B) TWO MAIN TYPES OF COMBINATIONS -- HIERARCHICAL AND CROSS-PRODUCT -- ARE ANALYZED, CLARIFYING THE RELATION BETWEEN REPRESENTATIONS FOR COMPONENT DATA STRUCTURES AND A REPRESENTATION FOR THE COMBINATION. (A)

8P, 5R.

343 SOFTWARE ENGINEERING

ROSS, D.T., GOODENOUGH, J.B., & IRVINE, C.A. SOFTWARE ENGINEERING: PROCESS, PRINCIPLES, AND GOALS. COMPUTER MAGAZINE, MAY 1975, 8(5), 17-27.

DESCRIPTION:

THIS PAPER ATTEMPTS TO DEFINE THE PRINCIPLES AND GOALS THAT AFFECT THE PRACTICE OF SOFTWARE ENGINEERING. ITS INTENT IS TO ORGANIZE THESE ASPECTS OF SOFTWARE ENGINEERING INTO A FRAMEWORK THAT RATIONALIZES AND ENCOURAGES THEIR PROPER USE, WHILE PLACING IN PERSPECTIVE THE DIVERSITY OF TECHNIQUES, METHODS, AND TOOLS THAT PRESENTLY COMPRISE THE SUBJECT OF SOFTWARE ENGINEERING. (O)

11P, 20R.

344 SOFTWARE ENGINEERING

ROYCE, W.W. SOFTWARE REQUIREMENTS ANALYSIS: SIZING AND COSTING. IN E. HOROWITZ (ED.), PRACTICAL STRATEGIES FOR DEVELOPING LARGE SOFTWARE SYSTEMS, READING, MASSACHUSETTS: ADDISON-WESLEY, 1975, 57-71.

DESCRIPTION:

THE SUCCESS OR FAILURE OF A LARGE SOFTWARE PROJECT IS OFTEN DUE TO THE CONSISTENCY AND COMPLETENESS IN SPECIFYING SOFTWARE REQUIREMENTS AND THE ACCURACY WITH WHICH THESE REQUIREMENTS CAN BE TRANSLATED INTO COST AND SCHEDULE ESTIMATES AND HARDWARE REQUIREMENTS. THIS PAPER DISCUSSES FOUR TECHNIQUES FOR IMPROVING REQUIREMENTS ANALYSIS.

1. THERE SHOULD BE AUTOMATED TOOLS FOR ANALYZING THE COMPLETENESS, CONSISTENCY, ALLOCATION, AND TRACEABILITY OF THE REQUIREMENTS.

2. IMPROVED METHODS FOR DESIGNLESS COSTING TO QUANTIFY REQUIREMENTS SELECTION ARE REQUIRED.

3. PREVIOUSLY DEVELOPED SIMULATIONS, THAT CAN BE USED ON THE CURRENT PROBLEM WITHOUT ADDITIONAL CODING, SHOULD BE DEVELOPED.

4. EFFECTIVE SIMULATION-ORIENTED LANGUAGES TO AID IN SIMULATION BUILDING SHOULD BE IMPROVED TO BROADEN THEIR APPLICATIONS. (MEA)

15P, 11R.

345 PROGRAMMING LANGUAGES

RUBEY, R.J., ET AL. COMPARATIVE EVALUATION OF PL/1 (USAF REPORT NO. ESD-TR-68-150). SAN PEDRO, CALIFORNIA: LOGICON, INC., 1968. (NTIS NO. AD 669096)

DESCRIPTION:

SEVEN BENCHMARK PROBLEMS WERE EACH IMPLEMENTED TWICE BY THE SAME PROGRAMMER, ONCE IN PL/1 AND ONCE IN ANOTHER HIGHER LEVEL LANGUAGE (COBOL, FORTRAN, OR JOVIAL) APPROPRIATE TO THE APPLICATION AREA REPRESENTED BY THE PROBLEM. OVERALL, IT WAS FOUND THAT PL/1 HAD ADVANTAGES OVER BOTH FORTRAN AND JOVIAL AND WAS ABOUT EQUAL TO COBOL FOR THE RESPECTIVE APPLICATION AREAS. THE QUANTITATIVE DATA OBTAINED FROM THE IMPLEMENTATIONS GENERALLY INDICATED THAT THE PL/1 VERSIONS HAD FEWER STATEMENTS IN THE SOURCE PROGRAMS AND WERE CODED MORE RAPIDLY THAN THEIR COMPARISON-LANGUAGE COUNTERPARTS, BUT TOOK LONGER TO DEBUG AND HAD A HIGHER FREQUENCY OF ERRORS. THE QUALITATIVE, SUBJECTIVE OPINIONS OF THE PROBLEM PROGRAMMERS AND PROJECT ANALYSTS INDICATED THAT PL/1 WAS GENERALLY SUPERIOR TO THE COMPARISON LANGUAGES WITH REGARD TO SUITABILITY FOR A WIDE RANGE OF PROBLEMS, NATURALNESS, GENERALITY, AND EASE OF USE. INEFFICIENCIES OBSERVED IN THE LANGUAGE COMPILERS AND ASSOCIATED OPERATING SYSTEMS UTILIZED FOR THE BENCHMARK PROBLEMS INDICATED THAT IMPROVEMENTS ARE REQUIRED IN THESE AREAS IF THE BENEFITS OBTAINABLE FROM THE USE OF HIGHER LEVEL LANGUAGES ARE TO BE FULLY REALIZED. (A)

28?P, OR.

346 PROGRAMMING, ERRORS

RUBEY, R.J., DANA, J.A., & BICHE, P.W. QUANTITATIVE ASPECTS OF SOFTWARE VALIDATION. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 150-155 (A SIMILAR PAPER IS RUBEY, R.J. QUANTITATIVE ASPECTS OF SOFTWARE VALIDATION. PROCEEDINGS, INTERNATIONAL CONFERENCE ON RELIABLE SOFTWARE, 21-23 APRIL 1975, LOS ANGELES, CALIFORNIA. SIGPLAN NOTICES, JUNE 1975, 10(6), 246-251).

DESCRIPTION:

THIS PAPER DISCUSSES THE NEED FOR QUANTITATIVE DESCRIPTIONS OF SOFTWARE ERRORS AND METHODS FOR GATHERING SUCH DATA. THE SOFTWARE DEVELOPMENT CYCLE IS REVIEWED, AND THE FREQUENCY OF THE ERRORS THAT ARE DETECTED DURING SOFTWARE DEVELOPMENT AND INDEPENDENT VALIDATION ARE COMPARED. DATA OBTAINED FROM VALIDATION EFFORTS ARE PRESENTED, INDICATING THE NUMBER OF ERRORS IN TEN CATEGORIES AND THREE SEVERITY LEVELS; THE INFERENCES THAT CAN BE DRAWN FROM THESE DATA ARE DISCUSSED. DATA DESCRIBING THE EFFECTIVENESS OF VALIDATION TOOLS AND TECHNIQUES AS A FUNCTION OF TIME ARE PRESENTED AND DISCUSSED. THE SOFTWARE VALIDATION COST IS CONTRASTED WITH THE SOFTWARE DEVELOPMENT COST. THE APPLICATIONS OF BETTER QUANTITATIVE SOFTWARE ERROR DATA ARE SUMMARIZED. (A)

6P, OR.

347 PROGRAMMING LANGUAGES

RULIFSON, J.F., WALDINGER, R.J., & DERKSEN, J.A. A LANGUAGE FOR WRITING PROBLEM-SOLVING PROGRAMS. IN PROCEEDINGS IFIP CONGRESS 1971 (PRESENTED AT LJUBLJANA, YUGOSLAVIA), AUGUST 1971.

348 ARTIFICIAL INTELLIGENCE

RUTH, G.R. INTELLIGENT PROGRAM ANALYSIS. ARTIFICIAL INTELLIGENCE, 1976, 7, 65-85.

DESCRIPTION:

IN ORDER TO EXAMINE THE POSSIBILITIES OF USING A COMPUTER AS AN AID TO TEACHING PROGRAMMING, A PROTOTYPE INTELLIGENT PROGRAM ANALYZER HAS BEEN CONSTRUCTED. ITS DESIGN ASSUMES THAT A SYSTEM CANNOT ANALYZE A PROGRAM UNLESS IT CAN "UNDERSTAND" IT; UNDERSTANDING BEING BASED ON A KNOWLEDGE OF WHAT MUST BE ACCOMPLISHED AND HOW CODE IS USED TO EXPRESS THE INTENTIONS.

IT WAS FOUND THAT A ONE-PAGE DESCRIPTION OF TWO COMMON SORTING ALGORITHMS OR OF SOME COMMON APPROXIMATION PROBLEMS WAS SUFFICIENT FOR THE COMPUTER TO UNDERSTAND AND ANALYZE A WIDE VARIETY OF PROGRAMS AND IDENTIFY AND DESCRIBE ALMOST ALL ERRORS. (A)

349 PROGRAMMING LANGUAGES

RYCHENER, M.D. CONTROL REQUIREMENTS FOR THE DESIGN OF PRODUCTION SYSTEM ARCHITECTURES. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 37-44 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 37-44).

DESCRIPTION:

PROGRAMS IN THE ARTIFICIAL INTELLIGENCE DOMAIN IMPOSE UNUSUAL REQUIREMENTS ON CONTROL STRUCTURES. PRODUCTION SYSTEMS ARE A CONTROL STRUCTURE WITH PROMISING ATTRIBUTES FOR BUILDING GENERALLY INTELLIGENT SYSTEMS WITH LARGE KNOWLEDGE BASES. THIS PAPER PRESENTS EXAMPLES TO ILLUSTRATE THE UNUSUAL POSITION TAKEN BY PRODUCTION SYSTEMS ON A NUMBER OF CONTROL AND PATTERN-MATCHING ISSUES. EXAMPLES ARE CHOSEN TO ILLUSTRATE CERTAIN POWERFUL FEATURES AND TO PROVIDE CRITICAL TESTS WHICH MIGHT BE USED TO EVALUATE THE EFFECTIVENESS OF NEW DESIGNS. (A)

8P, 12R.

350 COMMENTS

SACHS, J. SOME COMMENTS ON COMMENTS. ACM/SIGDOC SYSTEMS DOCUMENTATION NEWSLETTER, 1976, 3(7), 7-14.

DESCRIPTION:

PROGRAMMERS PROBABLY SPEND MORE TIME WRITING AND READING COMMENTS THAN ANY OTHER KIND OF DOCUMENTATON. THUS, IT IS STRANGE THAT LITTLE ATTENTION HAS BEEN GIVEN TO THE PROBLEM OF MAKING COMMENTS AS USEFUL AS POSSIBLE. (A)

8P, 3R.

351 GENERAL DISCUSSION

SACKMAN, H. COMPUTERS, SYSTEM SCIENCE, AND EVOLVING SOCIETY. NEW YORK, NEW YORK: WILEY, 1967.

352 TIME-SHARING VS BATCH

SACKMAN, H. EXPERIMENTAL INVESTIGATION OF USER PERFORMANCE IN TIME-SHARED COMPUTING SYSTEMS: RETROSPECT, PROSPECT, AND THE PUBLIC INTEREST (TECHNICAL REPORT NO. SP-2846). SANTA MONICA, CALIFORNIA: SYSTEM DEVELOPMENT CORPORATION, 1967.

353 COMPARISONS OF TIME-SHARING WITH BATCH PROCESSING

SACKMAN, H. TIME-SHARING VERSUS BATCH PROCESSING: THE EXPERIMENTAL EVIDENCE. AFIPS CONFERENCE PROCEEDINGS, 1968, 32, 1-10 (ALSO PUBLISHED WITH ADDITIONAL SUMMARY SECTION AS TECHNICAL REPORT SP-2975, SYSTEM DEVELOPMENT CORP., SANTA MONICA, CALIFORNIA, OCTOBER 1967, NTIS NO. AD 661665).

DESCRIPTION:

THE CONTINUING CONTROVERSY OVER THE RELATIVE MERITS OF TIME-SHARING VERSUS BATCH PROCESSING HAS TAKEN A NEW AND SIGNIFICANT TURN FROM PREDISCIPLINARY SPECULATION TO APPLIED SCIENTIFIC EXPERIMENTATION. WITHIN THE LAST TWO YEARS, FIVE EXPERIMENTAL STUDIES HAVE APPEARED IN THE LITERATURE, EACH COMPARING SOME FORM OF ONLINE AND OFFLINE DATA PROCESSING WITH RESPECT TO MAN-MACHINE MEASURES OF SYSTEM PERFORMANCE. THESE FIVE PIONEERING STUDIES COMPRISE THE FIRST SUBSTANTIVE DATA BASE FOR COMPARING AND EVALUATING EXPERIMENTAL METHODOLOGY AND FINDINGS BEARING ON THE GROWING AND CHANGING COMPETITION BETWEEN TIME-SHARING AND BATCH PROCESSING SYSTEMS. THIS PAPER PROVIDES A CRITICAL REVIEW OF THESE FIVE EXPERIMENTS, SUMMARIZES FINDINGS, PROBLEMS AND PITFALLS, AND OFFERS RECOMMENDATIONS FOR FUTURE EXPERIMENTAL WORK. (A)

FIVE EXPERIMENTAL COMPARISONS OF TIME-SHARING AND BATCH SYSTEMS ARE REVIEWED. THE COMPOSITE RESULTS INDICATE THAT TIME-SHARING REQUIRES FEWER MAN-HOURS, POSSIBLY PRODUCES HIGHER QUALITY RESULTS, AND IS PREFERRED BY USERS. BATCH PROCESSING MAY REQUIRE LESS COMPUTER TIME AND INVOLVE LOWER COSTS. THE METHODOLOGICAL ISSUES OF SUCH COMPARISONS ARE DISCUSSED AND SUGGESTIONS ARE MADE FOR ADDITIONAL EXPERIMENTS. (MEA)

10P, 12R.

354 USER PERFORMANCE IN TIME-SHARING AND BATCH SYSTEMS

SACKMAN, H. EXPERIMENTAL ANALYSIS OF MAN-COMPUTER PROBLEM-SOLVING. HUMAN FACTORS, 1970, 12, 187-201.

DESCRIPTION:

EXPERIMENTAL METHODS AND FINDINGS IN HUMAN PROBLEM-SOLVING USING ON-LINE AND OFF-LINE COMPUTER SYSTEMS ARE REVIEWED. THE ADVENT OF TIME-SHARING SYSTEMS IN THE LAST DECADE PRODUCED AN INITIAL BODY OF EMPIRICAL DATA FROM USER STATISTICS AND EXPERIMENTAL STUDIES COMPARING TIME-SHARING WITH BATCH-PROCESSING. THIS BODY OF DATA IS REVIEWED FOR ITS IMPLICATIONS TO THE CONTROVERSY OVER BATCH AND TIME-SHARING SYSTEMS AND TO THE UNDERSTANDING OF HUMAN BEHAVIOR IN THE MAN-COMPUTER SETTING. A PLEA IS MADE FOR INTERDISCIPLINARY CROSS-FERTILIZATION BETWEEN BEHAVIORAL AND COMPUTER SCIENCES TO BRIDGE THE HUMANISTIC LAG IN MAN-COMPUTER COMMUNICATION. (A)

15P, 20R.

355 COMPARISON OF TIME-SHARING WITH BATCH PROCESSING

SACKMAN, H. MAN-COMPUTER PROBLEM SOLVING. PRINCETON, NEW JERSEY: AUERBACH, 1970.

DESCRIPTION:

THIS BOOK IS CONCERNED WITH THE GROWING EXPERIMENTAL EVIDENCE ON MAN-COMPUTER PROBLEM SOLVING, PARTICULARLY IN THE COMPETITION BETWEEN TIME-SHARING AND BATCH-PROCESSING COMPUTER SYSTEMS. THE BOOK IS DIVIDED INTO FOUR PARTS. PART I ESSENTIALLY CONSISTS OF AN INTRODUCTION TO TIME-SHARING AND BATCH-PROCESSING, THE HISTORICAL BACKGROUND OF THE SCIENTIFIC STUDY OF THE HUMAN USE OF COMPUTERS, AND SUMMARY ACCOUNTS OF EXPLORATORY ONLINE/OFFLINE STUDIES THAT PRECEDED THE COMPREHENSIVE STUDIES DESCRIBED IN PARTS II AND III. THE EPILOGUE, PART IV, WAS DESIGNED TO PULL TOGETHER AND SUMMARIZE THE VARIOUS STRANDS OF RESEARCH ON MAN-COMPUTER PROBLEM SOLVING UNDER ONLINE AND OFFLINE CONDITIONS AS REPORTED IN THIS BOOK. SPECIAL EMPHASIS IS PLACED ON THE INTERFACE BETWEEN THESE ONLINE/OFFLINE STUDIES AND THE MAINSTREAM OF THE BEHAVIORAL LITERATURE ON HUMAN PROBLEM SOLVING. THE BOOK CONCLUDES WITH A PREVIEW OF MASS COMPUTER UTILITIES, AND A PLEA FOR COOPERATIVE INTERDISCIPLINARY RESEARCH ON EXPERIMENTAL COMMUNITY PROTOTYPES TO MEET THE CHALLENGE OF THE PUBLIC INTEREST IN THE COMPUTER-SERVICED SOCIETY OF THE FUTURE. (A, ABBR.)

288P, 69R.

356 COMPARISON OF TIME-SHARING AND BATCH PROCESSING
SACKMAN, H., ERIKSON, W.J., & GRANT, E.E. EXPLORATORY EXPERIMENTAL STUDIES COMPARING ONLINE AND OFFLINE PROGRAMMING PERFORMANCE. COMMUNICATIONS OF THE ACM, 1968, 11, 3-11.
DESCRIPTION:
TWO EXPLORATORY EXPERIMENTS WERE CONDUCTED AT SYSTEM DEVELOPMENT CORPORATION TO COMPARE DEBUGGING PERFORMANCE OF PROGRAMMERS WORKING UNDER CONDITIONS OF ONLINE AND OFFLINE ACCESS TO A COMPUTER. THESE ARE THE FIRST KNOWN STUDIES THAT MEASURE PROGRAMMERS' PERFORMANCE UNDER CONTROLLED CONDITIONS FOR STANDARD TASKS.

STATISTICALLY SIGNIFICANT RESULTS OF BOTH EXPERIMENTS INDICATED FASTER DEBUGGING UNDER ONLINE CONDITIONS, BUT PERHAPS THE MOST IMPORTANT PRACTICAL FINDING INVOLVES THE STRIKING INDIVIDUAL DIFFERENCES IN PROGRAMMER PERFORMANCE. METHODOLOGICAL PROBLEMS ENCOUNTERED IN DESIGNING AND CONDUCTING THESE EXPERIMENTS ARE DESCRIBED; LIMITATIONS OF THE FINDINGS ARE POINTED OUT; HYPOTHESES ARE PRESENTED TO ACCOUNT FOR RESULTS; AND SUGGESTIONS ARE MADE FOR FURTHER RESEARCH. (A)
9P, 15R.

357 COMPARISON OF TIME-SHARING AND BATCH PROCESSING
SACKMAN, H., & GOLD, M.M. TIME-SHARING VERSUS BATCH PROCESSING: AN EXPERIMENTAL INQUIRY INTO HUMAN PROBLEM-SOLVING (TECHNICAL REPORT NO. SP-3110). SANTA MONICA, CALIFORNIA: SYSTEM DEVELOPMENT CORP., 1968.

358 COMPILER TESTING
SAMET, H. A NORMAL FORM FOR COMPILER TESTING. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 155-162 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 155-162).
DESCRIPTION:
A FORMALISM IS PRESENTED FOR OBTAINING A NORMAL FORM TO BE USED IN REPRESENTING PROGRAMS FOR COMPILER TESTING. EXAMPLES ARE USED TO MOTIVATE THE FEATURES THAT MUST BE CONSIDERED WHEN DEVELOPING SUCH A FORMALISM. IT IS PARTICULARLY SUITABLE FOR HEURISTICALLY OPTIMIZED CODE AND HAS BEEN SUCCESSFULLY USED IN A SYSTEM FOR PROVING THAT PROGRAMS WRITTEN IN A SUBSET OF LISP ARE CORRECTLY TRANSLATED TO ASSEMBLY LANGUAGE. (A)
8P, 19R.

359 PROGRAMMING LANGUAGES, REVIEW
SAMMET, J.E. PROGRAMMING LANGUAGES: HISTORY AND FUNDAMENTALS. ENGLEWOOD CLIFFS, NEW JERSEY: PRENTICE-HALL, 1969.
DESCRIPTION:
THE PRIMARY PURPOSE OF THIS BOOK IS TO SERVE AS A REFERENCE FOR AN OVERALL VIEW OF HIGHER LEVEL LANGUAGES. THIS BOOK BRINGS TOGETHER IN ONE PLACE, AND IN A CONSISTENT FASHION, FUNDAMENTAL INFORMATION ON PROGRAMMING LANGUAGES, INCLUDING HISTORY, GENERAL CHARACTERISTICS, SIMILARITIES, AND DIFFERENCES.

A SECOND PURPOSE OF THE BOOK IS TO PROVIDE SPECIFIC BASIC INFORMATION ON ALL THE SIGNIFICANT, AND MOST OF THE MINOR, HIGHER LEVEL LANGUAGES DEVELOPED IN THE UNITED STATES.

THE THIRD PURPOSE OF THE BOOK IS TO PROVIDE HISTORY AND PERSPECTIVE FOR THIS PARTICULAR ASPECT OF THE PROGRAMMING FIELD. BECAUSE OF THE RAPIDLY CHANGING NATURE OF THIS TYPE OF WORK, NEW LANGUAGES APPEAR DAILY (LITERALLY) AND SO THIS BOOK REPRESENTS A SNAPSHOT OF - AND AN (INDIRECT) EXPLANATION OF HOW WE ARRIVED AT - THE SITUATION AT A GIVEN POINT IN TIME, NAMELY THE FALL OF 1967.

OTHER PURPOSES ARE TO PROVIDE AN EXTENSIVE BIBLIOGRAPHY OF RELEVANT MATERIAL, TO SHOW VARIOUS PHILOSOPHIES OF LANGUAGE DESIGN, TO DESCRIBE A NUMBER OF KEY FACTORS INVOLVED IN CHOOSING A LANGUAGE, AND TO PROVIDE THE READER WITH ENOUGH INFORMATION SO THAT HE CAN DECIDE WHICH LANGUAGES HE WISHES TO EXAMINE IN DETAIL. (A, ABBR)

360 PROGRAMMING LANGUAGES
SAMMET, J.E. PROBLEMS IN, AND A PRAGMATIC APPROACH TO, PROGRAMMING LANGUAGE MEASUREMENT. AFIPS CONFERENCE PROCEEDINGS, 1971.

361 PROGRAMMING LANGUAGES
SAMMET, J.E. PROGRAMMING LANGUAGES: HISTORY AND FUTURE. COMMUNICATIONS OF THE ACM, 1972, 15, 601-610.

362 ROSTER OF PROGRAMMING LANGUAGES
SAMMET, J.E. ROSTER OF PROGRAMMING LANGUAGES FOR 1976-77. SIGPLAN NOTICES, NOVEMBER 1978, 13(11), 56-85.

363 COMPARISON OF TIME-SHARING WITH BATCH PROCESSING
SCHATZOFF, M., TSAO, R., & WIIG, R. AN EXPERIMENTAL COMPARISON OF TIME SHARING AND BATCH PROCESSING. COMMUNICATIONS OF THE ACM, 1967, 10, 261-265.
DESCRIPTION:
THE EFFECTIVENESS FOR PROGRAM DEVELOPMENT OF THE MIT COMPATIBLE TIME-SHARING SYSTEM (CTSS) WAS COMPARED WITH THAT OF THE IBM IBSYS BATCH PROCESSING SYSTEM BY MEANS OF A STATISTICALLY DESIGNED EXPERIMENT. AN IDENTICAL SET OF FOUR PROGRAMMING PROBLEMS WAS ASSIGNED TO EACH OF A GROUP OF FOUR PROGRAMMING SUBJECTS. DATA WAS OBTAINED FOR SIX VARIABLES WHICH WERE CONSIDERED TO BE DEFINITIVE OF "SYSTEM EFFECTIVENESS", AND ANALYSIS OF VARIANCE TECHNIQUES WERE EMPLOYED TO ESTIMATE SYSTEM DIFFERENCES IN THESE VARIABLES. ANALYSIS OF THE RESULTS PROVIDED STRONG EVIDENCE OF IMPORTANT SYSTEM DIFFERENCES. (A) 5P, 2R.

364 TIME-SHARING
SCHERR, A.L., AN ANALYSIS OF TIME-SHARED COMPUTER SYSTEMS (RESEARCH MONOGRAPH NO. 36). CAMBRIDGE, MASSACHUSETTS: THE M. I. T. PRESS, 1967.

365 SOFTWARE TESTING
SCHLENDER, P. APPLICATION OF DISCIPLINED SOFTWARE TESTING. IN R. RUSTIN (ED.), DEBUGGING TECHNIQUES IN LARGE SYSTEMS. ENGLEWOOD CLIFFS, NEW JERSEY: PRENTICE-HALL, 1971, 141-142.

366 SOFTWARE ERRORS

SCHNEIDEWIND, N.F. ANALYSIS OF ERROR PROCESSES IN COMPUTER SOFTWARE. IN PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON RELIABLE SOFTWARE, 21-23 1975, LOS ANGELES, CALIFORNIA. SIGPLAN NOTICES, 1975, 10, 337-346.

DESCRIPTION:

A NON-HOMOGENEOUS POISSON PROCESS IS USED TO MODEL THE OCCURRENCE OF ERRORS DETECTED DURING FUNCTIONAL TESTING OF COMMAND AND CONTROL SOFTWARE. THE PARAMETERS OF THE DETECTION PROCESS ARE ESTIMATED BY USING A COMBINATION OF MAXIMUM LIKELIHOOD AND WEIGHTED LEAST SQUARES METHODS. ONCE PARAMETER ESTIMATES ARE OBTAINED, FORECASTS CAN BE MADE OF CUMULATIVE NUMBER OF DETECTED ERRORS. FORECASTING EQUATIONS OF CUMULATIVE CORRECTED ERRORS, ERRORS DETECTED BUT NOT CORRECTED, AND THE TIME REQUIRED TO DETECT OR CORRECT A SPECIFIED NUMBER OF ERRORS, ARE DERIVED FROM THE DETECTED ERROR FUNCTION. THE VARIOUS FORECASTS PROVIDE DECISION AIDS FOR MANAGING SOFTWARE TESTING ACTIVITIES. NAVAL TACTICAL DATA SYSTEM SOFTWARE ERROR DATA ARE USED TO EVALUATE SEVERAL VARIATIONS OF THE FORECASTING EQUATIONS. BECAUSE OF CHANGES WHICH TAKE PLACE IN THE ACTUAL DETECTED ERROR PROCESS, IT WAS FOUND THAT RECENT ERROR OBSERVATIONS ARE MORE REPRESENTATIVE OF FUTURE ERROR OCCURRENCES THAN ARE EARLY OBSERVATIONS. BASED ON A LIMITED TEST OF THE MODEL, ACCEPTABLE ACCURACY WAS OBTAINED WHEN USING THE PREFERRED FORECASTING METHOD.

10P, 7R.

367 SOFTWARE COMPLEXITY

SCHNEIDEWIND, N.F., & GREEN, T.F. SIMULATION OF ERROR DETECTION IN COMPUTER PROGRAMS. SIMULETTER, APRIL 1976, 7(3), 8-12.

DESCRIPTION:

THE RELATIONSHIP BETWEEN COMPUTER PROGRAM COMPLEXITY AND ERROR DETECTION CAPABILITY IS INVESTIGATED BY REPRESENTING A PROGRAM AS A DIRECTED GRAPH AND SIMULATING THE DETECTION AND CORRECTION OF ERRORS. VARIABLES OF INTEREST ARE TEST COVERAGE, NUMBER OF INPUTS, RESIDUAL ERRORS, EXECUTION TIME, CORRECTION TIME AND NODE-ARC-LOOP RELATIONSHIPS. ONE APPLICATION IS IN SOFTWARE DESIGN WHERE THE INFORMATION PROVIDED BY THE MODEL WOULD BE USED TO SELECT PROGRAM STRUCTURES WHICH ARE EASY TO TEST. A SECOND APPLICATION IS IN SOFTWARE TESTING WHERE TEST STRATEGIES AND ALLOCATION OF TEST EFFORTS WOULD BE BASED ON ERROR DETECTION AND COMPLEXITY CONSIDERATIONS. (A)

5P, 3R.

368 PROGRAMMER PERFORMANCE

SCOTT, R.F. A PROGRAMMER PRODUCTIVITY PREDICTION MODEL (UNPUBLISHED DOCTORAL DISSERTATION, TEXAS A&M UNIVERSITY). 1973.

DESCRIPTION:

A COMPUTER PROGRAMMER PRODUCTIVITY PREDICTION MODEL HAS BEEN DEVELOPED. THIS DEVELOPMENT WAS PRECEDED BY AN INVESTIGATION OF THE PROGRAMMER PRODUCTIVITY FACTORS. THE RESEARCH WAS ACCOMPLISHED IN THREE PARTS.

TO IDENTIFY IMPORTANT PRODUCTIVITY FACTORS, A DATA ANALYSIS WAS PERFORMED USING A COMBINATION OF EXISTING DATA BASES WITH STEPWISE MULTIVARIATE REGRESSION ANALYSIS. THE DERIVED EQUATION, WITH PROGRAMMER PRODUCTIVITY AS THE DEPENDENT VARIABLE, IS INCLUDED.

TO GAIN ADDITIONAL INSIGHT INTO PRODUCTIVITY FACTORS, AN ALTERNATE NON-ANALYTICAL METHOD WAS USED. EXPERTS ON COMPUTER PROGRAMMING PROJECT MANAGEMENT WERE SURVEYED. THE ITERATIVE DELPHI SURVEY TECHNIQUE WAS USED TO IDENTIFY IMPORTANT PROGRAMMER PRODUCTIVITY PARAMETERS FROM A PANEL OF EXPERTS. A BACKGROUND REPORT ON THE DELPHI PROCEDURE AND THE SURVEY RESULTS ARE INCLUDED. THE IMPORTANCE OF PROGRAMMER INTERACTIONS WAS EVIDENT IN BOTH THE DATA ANALYSIS AND DELPHI STUDIES.

MODELING WAS NEXT CONSIDERED. THE OBJECTIVE WAS TO FURTHER STUDY PROJECT COMMUNICATIONS AND PROVIDE A PRODUCTIVITY PREDICTION ABILITY. A DISCUSSION OF SMALL GROUP, INFORMATION FEEDBACK AND MULTIPROCESSOR MODEL CONCEPTS IS PRESENTED. A PROJECT MODEL USING THE MULTIPROCESSOR CONCEPT IS PROGRAMMED IN A PROGRAMMING LANGUAGE (APL). THE MODEL EMPLOYS THE CONCEPT OF PROGRAMMER ACTIVITY PROFILES AND USES A CUMULATIVE EXPONENTIAL PRODUCTIVITY RATE. THREE EXAMPLE EXPERIMENTS ARE DISCUSSED.

ALL RESULTS EMPHASIZE THE IMPORTANCE OF INTERACTION AMONG PROGRAMMERS IN REDUCING PRODUCTIVITY. NUMEROUS RESULTS ARE INCLUDED ON THE RELATIVE IMPORTANCE OF VARIOUS VARIABLES TO PRODUCTIVITY. COMMUNICATIONS PATTERNS, NUMBER OF PROGRAMMERS, LENGTH OF PROJECT AND INDIVIDUAL PRODUCTIVITY VARIABLES ARE USED IN THE MODEL FOR PRODUCTIVITY PREDICTION.

SOME RECOMMENDATIONS AND EXTENSIONS FOR FURTHER RESEARCH ARE PRESENTED. (A)

136P, 82R.

369 PROGRAMMER PERFORMANCE

SCOTT, R.F., & SIMMONS, D.B. PROGRAMMER PRODUCTIVITY AND THE DELPHI TECHNIQUE. DATAMATION, MAY 1974, 20(5), 71-73.

DESCRIPTION:

THE DELPHI TECHNIQUE WAS USED TO DETERMINE WHICH VARIABLES PROGRAMMING PROJECT MANAGERS FEEL ARE RELATED TO PROGRAMMER PRODUCTIVITY. THIS TECHNIQUE INVOLVES ANONYMOUS RESPONSE, ITERATION, CONTROLLED FEEDBACK, AND STATISTICAL GROUP RESPONSE. THE RESULTS INDICATE THE IMPORTANCE OF PROVIDING THE INDEPENDENT PROGRAMMER WITH A WELL DOCUMENTED, THOROUGHLY DEFINED INDEPENDENT TASK, AND USING EXPERIENCED PROGRAMMERS WORKING IN HIGH-LEVEL LANGUAGES. STRUCTURED PROGRAMMING TECHNIQUES WERE CONSIDERED LESS IMPORTANT THAN DOCUMENTATION, PROGRAMMING TOOLS, AND EXPERIENCE. (MEA)

3P, 9R.

370 PROGRAMMER PERFORMANCE

SCOTT, R.F., & SIMMONS, D.B. PREDICTING PROGRAMMING GROUP PRODUCTIVITY: A COMMUNICATIONS MODEL. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, 1, 411-414. (ALSO: PROGRAMMER PRODUCTIVITY AND THE DELPHI TECHNIQUE. DATAMATION, 1974, 20, 71-73.)

DESCRIPTION:

METHODS OF STUDYING PROGRAMMER PRODUCTIVITY ARE DIFFICULT TO FIND. THE CLASSICAL METHODS OF OBSERVATION AND STATISTICAL ANALYSIS ARE IN MANY CASES INAPPROPRIATE. THIS PAPER DESCRIBES A SIMULATION APPROACH IN WHICH PROGRAMMERS ARE CONSIDERED TO BE INDIVIDUAL PROCESSORS. RELATIVE GROUP PRODUCTIVITY IS THEN MEASURED BASED UPON THE PRODUCTIVITY LEVELS AND COMMUNICATIONS RELATIONSHIPS OF THE PROCESSORS. (A)

371 NATURAL-LANGUAGE PROGRAMMING

SHAPIRO, S.C., & KWANSY, S.C. INTERACTIVE CONSULTING VIA NATURAL LANGUAGE (TECHNICAL REPORT NO. 12). BLOOMINGTON, INDIANA: INDIANA UNIVERSITY, COMPUTER SCIENCE DEPARTMENT, 1974.

DESCRIPTION:

INTERACTIVE PROGRAMMING SYSTEMS OFTEN CONTAIN HELP COMMANDS TO GIVE THE PROGRAMMER ON-LINE INSTRUCTION REGARDING THE USE OF THE VARIOUS SYSTEMS COMMANDS. WE ARGUE THAT IT WOULD BE RELATIVELY EASY TO MAKE THESE HELP COMMANDS SIGNIFICANTLY MORE HELPFUL BY HAVING THEM ACCEPT REQUESTS IN NATURAL LANGUAGE. AS A DEMONSTRATION, WE HAVE PROVIDED WEIZENBAUM'S ELIZA PROGRAM WITH A SCRIPT THAT TURNS IT INTO A NATURAL LANGUAGE SYSTEM CONSULTANT. (A)

32P, 13R.

372 SOFTWARE COMPLEXITY

SHELL, R.L. WORK MEASUREMENT FOR COMPUTER PROGRAMMING OPERATION. INDUSTRIAL ENGINEERING, 1972, 4(10), 32-36.

DESCRIPTION:

PRESENTING A METHODOLOGY TO DEVELOP WORK MEASUREMENT STANDARDS FOR COMPUTER PROGRAMMING OPERATIONS. COVERS FACTORS THAT AFFECT PROGRAMMING TIME, PROCEDURAL METHODS USED TO WRITE PROGRAMS, AND RELATIONSHIPS BETWEEN PROGRAM SIZE, COMPLEXITY, AND TIME. (A)

5P, 14R.

373 FACTORS AFFECTING MAINTAINANCE PROGRAMMING PERFORMANCE

SHEPPARD, S.B., BORST, M.A., CURTIS, B., & LOVE, L.T. PREDICTING PROGRAMMERS' ABILITY TO MODIFY SOFTWARE (TECHNICAL REPORT TR-388100-3). ARLINGTON, VIRGINIA: GENERAL ELECTRIC COMPANY, MAY 1978.

374 PROGRAM COMPREHENSION

SHEPPARD, S.B., BORST, M.A., & LOVE, L.T. PREDICTING SOFTWARE COMPREHENSIBILITY (TECHNICAL REPORT NO. TR-388100-2). ARLINGTON, VIRGINIA: GENERAL ELECTRIC COMPANY, FEBRUARY 1978. (NTIS NO. AD A051495)

375 SOFTWARE PHYSICS

SHEPPARD, S.B., & LOVE, L.T. A PRELIMINARY EXPERIMENT TO TEST INFLUENCES ON HUMAN UNDERSTANDING OF SOFTWARE (TECHNICAL REPORT TR-77-388100-1). ARLINGTON, VIRGINIA: GENERAL ELECTRIC, INFORMATION SYSTEMS PROGRAMS, JUNE 1977. (NTIS NO. AD A041916)

DESCRIPTION:

EIGHT EXPERIENCED PROGRAMMERS WERE EACH GIVEN THREE FORTRAN PROGRAMS TO MEMORIZE AND REPRODUCE FUNCTIONALLY, WITHOUT NOTES. THREE LEVELS OF COMPLEXITY OF CONTROL FLOW AND THREE LEVELS OF MNEMONIC VARIABLE NAMES WERE INDEPENDENTLY MANIPULATED. THE EXPERIMENTAL DESIGN WAS AN INCOMPLETE SPLIT-PLOT FACTORIAL WHERE EACH PROGRAMMER WAS GIVEN ONE VERSION OF EACH PROGRAM AND ALL LEVELS OF THE TWO PRIMARY INDEPENDENT VARIABLES.

THE PARTICIPANTS CORRECTLY RECALLED SIGNIFICANTLY MORE STATEMENTS WHEN THE COMPLEXITY OF CONTROL FLOW WAS REDUCED. DIFFERENCES IN RECALL FOR THE THREE LEVELS OF MNEMONIC VARIABLE NAMES WERE NOT SIGNIFICANT.

A FURTHER ANALYSIS COMPARED THE PERCENT OF STATEMENTS CORRECTLY RECALLED TO HALSTEAD'S E, A MEASURE OF THE EFFORT REQUIRED TO CODE A PROGRAM. THE PEARSON CORRELATION COEFFICIENT WAS -0.81, OVER THE 24 DATA POINTS; THUS INDICATING THAT HALSTEAD'S E IS A POWERFUL PREDICTOR OF ONE'S ABILITY TO UNDERSTAND A COMPUTER PROGRAM.

SEVERAL CHANGES IN THE EXPERIMENTAL DESIGN AND THE CONDUCT OF THE EXPERIMENT ITSELF ARE RECOMMENDED FOR FUTURE EXPERIMENTAL WORK IN THIS AREA. (A)

19P, 12R.

376 DATA STRUCTURES
SHNEIDERMAN, B. DATA STRUCTURES: DESCRIPTION, MANIPULATION, AND EVALUATION (UNPUBLISHED DOCTORAL DISSERTATION). STONY BROOK, NEW YORK: STATE UNIVERSITY OF NEW YORK AT STONY BROOK, DEPARTMENT OF COMPUTER SCIENCE, 1973.

377 PROGRAMMING LANGUAGES
SHNEIDERMAN, B. COGNITIVE PSYCHOLOGY AND PROGRAMMING LANGUAGE DESIGN. SIGPLAN NOTICES, JULY 1975, 10(7), 46-47.
DESCRIPTION:
PROGRAMMING LANGUAGE DESIGNERS CAN NO LONGER BE CONTENT WITH A THOROUGH KNOWLEDGE OF COMPUTER SCIENCE, BUT MUST BECOME FAMILIAR WITH THE IDEAS AND TECHNIQUES OF THE COGNITIVE PSYCHOLOGIST. COMMUNICATION BETWEEN COMPUTER SCIENTISTS AND COGNITIVE PSYCHOLOGISTS WILL BE HELPFUL IN THE DEVELOPMENT OF THE NEXT GENERATION OF PROGRAMMING LANGUAGES. IT WILL ALSO FACILITATE MORE WIDESPREAD COMPUTER LITERACY. (A)
2P, 1R.

378 PROGRAMMING LANGUAGES
SHNEIDERMAN, B. EXPERIMENTAL TESTING IN PROGRAMMING LANGUAGES, STYLISTIC CONSIDERATIONS AND DESIGN TECHNIQUES. AFIPS CONFERENCE PROCEEDINGS, 1975, 44, 653-656.
DESCRIPTION:
THIS PAPER BRIEFLY REVIEWS RESEARCH IN THE AREAS OF PROGRAMMING LANGUAGE DESIGN, STYLISTIC CONSIDERATIONS, AND PROGRAM DESIGN TECHNIQUES. A COMMON CRITICISM THAT CAN BE APPLIED TO THE MAJORITY OF THIS RESEARCH IS THE LACK OF CONTROLLED EXPERIMENTATION PRIOR TO THE IMPLEMENTATION OF NEW LANGUAGES AND TECHNIQUES. IN ORDER TO ESTABLISH AN APPROPRIATE FRAMEWORK FOR SUCH EXPERIMENTS, RELEVANT PROBLEM DOMAINS ARE DEFINED AND EXPERIMENTAL TECHNIQUES ARE DISCUSSED. (MEA)
4P, 30R.

379 PROGRAMMING
SHNEIDERMAN, B. EXPLORATORY EXPERIMENTS IN PROGRAMMER BEHAVIOR. INTERNATIONAL JOURNAL OF COMPUTER AND INFORMATION SCIENCES, 1976, 5, 123-143.
DESCRIPTION:
THE THESIS OF THIS PAPER IS THAT STUDIES OF PROGRAMMING SHOULD SEPARATE MACHINE-RELATED ISSUES FROM HUMAN FACTORS ISSUES. THE ISOLATION OF HUMAN FACTORS ISSUES WOULD ALLOW A MORE THOROUGH STUDY OF PROGRAMMING SINCE THE EXPERIMENTAL TECHNIQUES DEVELOPED BY COGNITIVE PSYCHOLOGISTS COULD BE READILY APPLIED. IN ORDER TO APPLY THESE TECHNIQUES, HOWEVER, IT IS FIRST NECESSARY TO DEFINE AN APPROPRIATE EXPERIMENTAL METHODOLOGY. THE METHODOLOGICAL ISSUES DISCUSSED IN THIS PAPER INCLUDE: CLASSIFICATION OF SUBJECTS, CLASSIFICATION OF PROGRAMS, RELEVANT TASKS THAT SHOULD BE STUDIED, AND MEASUREMENT TECHNIQUES.
THIS METHODOLOGY IS ILLUSTRATED IN TWO EXPERIMENTS. THE FIRST EXPERIMENT, ON PROGRAM MEMORIZATION, WAS MODELED AFTER A PARADIGM USED BY CHASE AND SIMON (COGNITIVE PSYCHOLOGY, 1973, 4, 55-81) IN A STUDY OF MEMORY FOR CHESS POSITIONS. CONSISTENT WITH RESULTS REPORTED IN COGNITIVE PSYCHOLOGY, IT WAS FOUND THAT EXPERIENCED PROGRAMMERS WERE ABLE TO RECALL SIGNIFICANTLY MORE STATEMENTS FROM A NORMAL-ORDER PROGRAM THAN INEXPERIENCED PROGRAMMERS, BUT THERE WERE NO DIFFERENCES IN RECALL FOR A SCRAMBLED-ORDER PROGRAM. THIS SUGGESTS THAT EXPERIENCED PROGRAMMERS HAVE LEARNED MORE EFFICIENT TECHNIQUES FOR RECODING PROGRAMS IN MEMORY. THE SECOND EXPERIMENT COMPARED THE BEHAVIOR OF SUBJECTS FROM TWO EXPERIENCE LEVELS IN COMPREHENDING PROGRAMS INVOLVING EITHER ARITHMETIC OR LOGICAL IF CONSTRUCTS, IN FORTRAN. THE RESULTS INDICATE THAT LOGICAL IF'S ARE INITIALLY EASIER TO COMPREHEND, BUT THAT THIS ADVANTAGE DISAPPEARS AS EXPERIENCE INCREASES. (MEA)
21P, 16R.

380 PROGRAMMING
SHNEIDERMAN, B. HUMAN FACTORS EXPERIMENTS IN PROGRAMMING: MOTIVATION, METHODOLOGY AND RESEARCH DIRECTIONS (TECHNICAL REPORT NO. ISM TR-9). COLLEGE PARK, MARYLAND: UNIVERSITY OF MARYLAND, DEPARTMENT OF INFORMATION SYSTEMS MANAGEMENT, SEPTEMBER 1976.

381 SOFTWARE DESIGN
SHNEIDERMAN, B. A REVIEW OF DESIGN TECHNIQUES FOR PROGRAMS AND DATA. SOFTWARE PRACTICE AND EXPERIENCE, 1976, 6, 555-567 (ALSO TECHNICAL REPORT NO. 25, INDIANA UNIVERSITY, BLOOMINGTON, INDIANA, APRIL 1975).
DESCRIPTION:
THE PROLIFERATION OF PAPERS ON PROGRAMMING METHODOLOGY FOCUS ON THE PROGRAM DEVELOPMENT PROCESS, BUT ONLY HINT AT THE FORM OF THE FINAL PROGRAM. THIS PAPER DISTINGUISHES BETWEEN THE DEVELOPMENT PROCESS AND THE PROGRAM PRODUCT AND PRESENTS A CATALOGUE OF POSSIBLE PROGRAM ORGANIZATIONS AND DATA STRUCTURES WITH EXAMPLES DRAWN FROM THE PUBLISHED LITERATURE. THE METHODS FOR SHARING DATA AMONG MODULES AND A CLASSIFICATION SCHEME FOR PROGRAMS AND DATA STRUCTURES IS PRESENTED. (A)
13P, 38R.

382 PROGRAM COMPREHENSION
SHNEIDERMAN, B. MEASURING COMPUTER PROGRAM QUALITY AND COMPREHENSION. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1977, 9, 465-478.

383 DATA BASE SYSTEMS
SHNEIDERMAN, B. IMPROVING THE HUMAN FACTORS ASPECT OF DATABASE INTERACTIONS. ACM TRANSACTIONS ON DATABASE SYSTEMS, IN PRESS, 1978.
DESCRIPTION:
THE WIDESPREAD DISSEMINATION OF COMPUTER AND INFORMATION SYSTEMS TO NON-TECHNICALLY TRAINED INDIVIDUALS REQUIRES A NEW APPROACH TO THE DESIGN AND DEVELOPMENT OF DATABASE INTERFACES. THIS PAPER PROVIDES THE MOTIVATIONAL BACKGROUND FOR CONTROLLED PSYCHOLOGICAL EXPERIMENTATION IN EXPLORING THE PERSON/MACHINE INTERFACE. FRAMEWORKS FOR THE REDUCTIONIST APPROACH ARE GIVEN, RESEARCH METHODS DISCUSSED, RESEARCH ISSUES PRESENTED AND A SMALL EXPERIMENT IS OFFERED AS AN EXAMPLE OF WHAT CAN BE ACCOMPLISHED. THIS EXPERIMENT IS A COMPARISON OF NATURAL AND ARTIFICIAL LANGUAGE QUERY FACILITIES. ALTHOUGH SUBJECTS POSED APPROXIMATELY EQUAL NUMBERS OF VALID QUERIES WITH EITHER FACILITY, NATURAL LANGUAGE USERS MADE SIGNIFICANTLY MORE INVALID QUERIES WHICH COULD NOT BE ANSWERED FROM THE DATABASE THAT WAS DESCRIBED. (A)

384 PROGRAMMING
SHNEIDERMAN, B. TEACHING PROGRAMMING: A SPIRAL APPROACH TO SYNTAX AND SEMANTICS. COMPUTERS AND EDUCATION 1, 1978, 3, 193-197 (ALSO TECHNICAL REPORT NO. ISM TR-15, COLLEGE PARK, MARYLAND: UNIVERSITY OF MARYLAND, DEPARTMENT OF INFORMATION SYSTEMS MANAGEMENT, FEBRUARY 1977).

385 PROGRAMMING

SHNEIDERMAN, B., & MAYER, R. TOWARDS A COGNITIVE MODEL OF PROGRAMMER BEHAVIOR (TECHNICAL REPORT NO. 37). BLOOMINGTON, INDIANA: INDIANA UNIVERSITY, COMPUTER SCIENCE DEPARTMENT, AUGUST 1975 (ALSO PUBLISHED AS SYNTACTIC/SEMANTIC INTERACTIONS IN PROGRAMMER BEHAVIOR: A MODEL AND EXPERIMENTAL RESULTS, INTERNATIONAL JOURNAL OF COMPUTER AND INFORMATION SCIENCE, IN PRESS).

DESCRIPTION:

THIS PAPER PRESENTS A COGNITIVE FRAMEWORK FOR DESCRIBING BEHAVIORS INVOLVED IN PROGRAM COMPOSITION, COMPREHENSION, DEBUGGING, MODIFICATION AND THE ACQUISITION OF NEW PROGRAMMING CONCEPTS, SKILLS AND KNOWLEDGE. AN INFORMATION PROCESS MODEL IS PRESENTED WHICH INCLUDES A LONG-TERM STORE OF SEMANTIC AND SYNTACTIC KNOWLEDGE, AND A WORKING MEMORY IN WHICH PROBLEM SOLUTIONS ARE CONSTRUCTED. NEW EXPERIMENTAL EVIDENCE IS PRESENTED TO SUPPORT THE MODEL. (A)

EXPERIMENTS ARE BRIEFLY CITED WHICH COMPARED LOGICAL WITH ARITHMETIC "IF" STATEMENTS, INVESTIGATED SEMANTIC EFFECTS ON MEMORY FOR PROGRAMS, AND INVESTIGATED THE EFFECTS OF COMMENTS, MEANINGFUL VARIABLE NAMES, MODULARIZATION OF PROGRAMS, AND FLOWCHARTS ON PROGRAM COMPREHENSION AND DEBUGGING. (HRR)

30P, 30R.

386 FLOWCHARTING, DOCUMENTATION

SHNEIDERMAN, B., MAYER, R., MCKAY, D., & HELLER, P. EXPERIMENTAL INVESTIGATIONS OF THE UTILITY OF FLOWCHARTS IN PROGRAMMING (TECHNICAL REPORT NO. 36). BLOOMINGTON, INDIANA: INDIANA UNIVERSITY, COMPUTER SCIENCE DEPARTMENT, AUGUST 1975 (ALSO: COMMUNICATIONS OF THE ACM, 1977, 20, 373-381).

DESCRIPTION:

THIS PAPER DESCRIBES PREVIOUS RESEARCH ON FLOWCHARTS AND A SERIES OF CONTROLLED EXPERIMENTS TO TEST THE UTILITY OF DETAILED FLOWCHARTS AS AN AID TO PROGRAM COMPOSITION, COMPREHENSION, DEBUGGING AND MODIFICATION. OUR RESULTS SHOWED NO STATISTICALLY SIGNIFICANT DIFFERENCE BETWEEN FLOWCHART AND NO FLOWCHART GROUPS, THEREBY CALLING INTO QUESTION THE UTILITY OF FLOWCHARTING. A PROGRAM OF FURTHER RESEARCH IS SUGGESTED. (A)

FIVE EXPERIMENTS ARE REPORTED: (1) SUBJECTS WHO PRODUCED FLOWCHARTS BEFORE WRITING A PROGRAM DID NO BETTER ON THE PROGRAMMING TASK THAN THOSE WHO DID ONLY THE LATTER; (2) WHEN FLOWCHARTS WERE PRESENTED WITH EITHER THE FIRST OR SECOND OF TWO PROGRAM COMPREHENSION TASKS, NO FLOWCHART-RELATED DIFFERENCES WERE OBSERVED; (3) WHEN TWO GROUPS OF SUBJECTS (ONE EXPERIENCED WITH FLOWCHARTS, ONE NOT) WERE GIVEN COMPREHENSION AND DEBUG TASKS WITH EITHER MACRO-FLOWCHARTS, MICRO-FLOWCHARTS, OR NO FLOWCHARTS, NO FLOWCHART MAIN EFFECTS WERE FOUND, BUT THOSE WITH PRIOR FLOWCHART EXPERIENCE PERFORMED BETTER WITH FLOWCHARTS (NOT STATISTICALLY SIGNIFICANT); (4) WITH AN EXPERIMENTAL DESIGN SIMILAR TO (3), BUT USING A MODIFICATION TASK, FLOWCHARTS HAD NO EFFECT ON SOLUTION TIME OR ERRORS; (5) SUBJECTS IN TWO COMPREHENSION TASKS DID BETTER (THOUGH NOT SIGNIFICANTLY) WITH THE PROGRAM ALONE THAN WITH PROGRAM AND FLOWCHART, OR FLOWCHART ALONE. (HRR)

53P, 17R.

387 DEBUGGING

SHNEIDERMAN, B., & MCKAY, D. EXPERIMENTAL INVESTIGATIONS OF COMPUTER PROGRAM DEBUGGING AND MODIFICATION. IN PROCEEDINGS OF THE 6TH CONGRESS OF THE INTERNATIONAL ERGONOMICS ASSOCIATION. SANTA MONICA, CA: HUMAN FACTORS SOCIETY, 1976, 557-563 (ALSO TECHNICAL REPORT NO. 48, COMPUTER SCIENCE DEPARTMENT, INDIANA UNIVERSITY, APRIL 1976).

DESCRIPTION:

ALTHOUGH GREATER EMPHASIS IS PLACED ON THE TASK OF COMPUTER PROGRAM COMPOSITION, DEBUGGING AND MODIFICATION OFTEN CONSUME MORE TIME AND EXPENSE IN PRODUCTION ENVIRONMENTS. DEBUGGING IS THE TASK OF LOCATING SYNTACTIC AND SEMANTIC ERRORS IN PROGRAMS AND CORRECTING THESE ERRORS. MODIFICATION IS THE CHANGE OF A WORKING PROGRAM TO PERFORM ALTERNATE TASKS.

THE FACTORS AND TECHNIQUES WHICH FACILITATE DEBUGGING AND MODIFICATIION ARE POORLY UNDERSTOOD, BUT ARE SUBJECT TO EXPERIMENTAL INVESTIGATION. CONTROLLED EXPERIMENTS CAN BE PERFORMED BY PRESENTING TWO AIDS AND REQUIRING THE SAME TASK. FOR EXAMPLE, IN ONE STUDY WE PRESENTED AN 81 LINE FORTRAN PROGRAM CONTAINING THREE BUGS TO DISTINCT GROUPS OF SUBJECTS. ONE OF THE GROUPS RECEIVED A DETAILED FLOWCHART, BUT OUR RESULTS INDICATED THAT THIS AID DID NOT FACILITATE THE DEBUGGING PROCEDURE. SIMILAR NEGATIVE RESULTS WERE OBTAINED FOR A MODIFICATION TASK.

IN OTHER EXPERIMENTS, COMMENTS AND MEANINGFUL VARIABLE NAMES WERE USEFUL IN DEBUGGING AND MODULARITY FACILITATED MODIFICATION. OTHER POTENTIALLY INFLUENTIAL FACTORS, WHICH ARE SUBJECT TO EXPERIMENTAL STUDY, INCLUDE INDENTATION RULES, TYPE OF CONTROL STRUCTURES, DATA STRUCTURE COMPLEXITY AND PROGRAM DESIGN.

THESE AND OTHER HUMAN FACTOR EXPERIMENTS IN PROGRAMMING HAVE LED TO A COGNITIVE MODEL OF PROGRAMMER BEHAVIOR WHICH DISTINGUISHES BETWEEN THE HIERARCHICALLY STRUCTURED, MEANINGFULLY ACQUIRED SEMANTIC KNOWLEDGE AND THE ROTELY MEMORIZED SYNTACTIC KNOWLEDGE. ERRORS CAN BE CLASSED INTO SYNTACTIC MISTAKES WHICH ARE RELATIVELY EASY TO LOCATE AND CORRECT AND TWO FORMS OF SEMANTIC MISTAKES. SEMANTIC ERRORS OCCUR WHILE CONSTRUCTING AN INTERNAL SEMANTIC STRUCTURE TO A REPRESENTATION IN THE SYNTAX OF A PROGRAMMING LANGUAGE. MODIFICATION IS INTERPRETED AS THE ACQUISITION OF AN INTERNAL SEMANTIC STRUCTURE BY STUDYING A PROGRAM, FOLLOWED BY MODIFICATION OF THIS STRUCTURE AND REVISION OF THE CODE. (A)

7P, 22R.

388 SOFTWARE ENGINEERING

SHOLL, H.A., & BOOTH, T.L. SOFTWARE PERFORMANCE MODELING USING COMPUTATION STRUCTURES. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 414-420.

389 PROGRAMMING ERRORS

SHOOMAN, M.L., & BOLSKY, M.I. TYPES, DISTRIBUTION, AND TEST AND CORRECTION TIMES FOR PROGRAMMING ERRORS. IN PROCEEDINGS OF THE 1975 INTERNATIONAL CONFERENCE ON SOFTWARE, SIGPLAN NOTICES, JUNE 1975, 10(6), 347-357.

DESCRIPTION:

IN ORDER TO DEVELOP SOME BASIC INFORMATION ON SOFTWARE ERRORS, AN EXPERIMENT IN COLLECTING DATA ON TYPES AND FREQUENCIES OF SUCH ERRORS WAS CONDUCTED AT BELL LABORATORIES.

THE PAPER REPORTS THE RESULTS OF THIS EXPERMENT, WHOSE OBJECTIVES WERE TO: (1) DEVELOP AND UTILIZE A SET OF TERMS FOR DESCRIBING POSSIBLE TYPES OF ERRORS, THEIR NATURE, AND THEIR FREQUENCY; (2) PERFORM A PILOT STUDY TO DETERMINE IF DATA OF THE TYPE REPORTED IN THIS PAPER COULD BE COLLECTED; (3) INVESTIGATE THE ERROR DENSITY AND ITS CORRESPONDENCE TO PREDICTIONS FROM PREVIOUS DATA REPORTED; (4) DEVELOP DATA ON HOW RESOURCES ARE EXPENDED IN DEBUGGING.

A PROGRAM OF APPROXIMATELY 4K MACHINE INSTRUCTIONS (FINAL SIZE) WAS CHOSEN. PROGRAMMERS WERE ASKED TO FILL OUT FOR EACH ERROR, IN ADDITION TO THE REGULAR TROUBLE REPORT/CORRECTION REPORT (TR/CR) FORM, A SPECIAL SUPPLEMENTARY TR/CR FORM FOR THE PURPOSES OF THIS EXPERIMENT. SIXTY-THREE TR/CR AND SUPPLEMENTARY FORMS WERE COMPLETED DURING THE TEST AND INTEGRATION PHASE OF THE PROGRAM.

IN GENERAL, THE DATA COLLECTED WERE FELT TO BE ACCURATE ENOUGH FOR THE PURPOSES OF THE ANALYSES PRESENTED. THE 63 FORMS REPRESENTED A LITTLE OVER 1-1/2% OF THE TOTAL NUMBER OF MACHINE INSTRUCTIONS OF THE PROGRAM (IN GOOD AGREEMENT WITH THE 1% TO 2% RANGE NOTED IN PREVIOUS STUDIES).

IT WAS DISCOVERED THAT A LARGE PERCENTAGE OF THE ERRORS WAS FOUND BY HAND PROCESSING (WITHOUT THE AID OF A COMPUTER). THIS METHOD WAS FOUND TO BE MUCH CHEAPER THAN TECHNIQUES INVOLVING MACHINE TESTING.

11P, 3R.

390 PROGRAMMING LANGUAGES

SIME, M.E. "SO I SAID IN THE MOST NATURAL WAY IF X=0 THEN BEGIN ..." THE EMPIRICAL STUDY OF COMPUTER LANGUAGES (MEMO NO. 132). SHEFFIELD, ENGLAND: UNIVERSITY OF SHEFFIELD, DEPARTMENT OF PSYCHOLOGY, UNDATED.

DESCRIPTION:

DESIGNERS OF COMPUTER LANGUAGES NEED EMPIRICAL STUDIES WHICH WILL SHOW HOW TO MAKE SYNTACTIC CONSTRUCTIONS EASY TO USE. THE RESULTS OF SOME EXPERIMENTS ON THE SYNTAX OF CONDITIONAL CONSTRUCTIONS, CARRIED OUT BY THE AUTHOR AND HIS COLLEAGUES, ARE DESCRIBED AND CONCLUSIONS ARE DRAWN FOR THE DESIGN OF PROCEDURAL LANGUAGES. (A)

36P, 15R.

391 PROGRAMMING LANGUAGES

SIME, M.E., ARBLASTER, A.T., & GREEN, T.R.G. REDUCING PROGRAMMING ERRORS IN NESTED CONDITIONALS BY PRESCRIBING A WRITING PROCEDURE. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1977, 9, 119-126.

DESCRIPTION:

WAYS TO REDUCE CARELESS PROGRAMMING ERRORS WERE INVESTIGATED. NON-PROGRAMMERS LEARNT TO WRITE NESTED CONDITIONAL PROGRAMS IN ONE OF THREE CONDITIONS: AN AUTOMATIC SYNTAX CONDITION, IN WHICH SYNTACTIC ERRORS WERE IMPOSSIBLE BECAUSE PROGRAMS WERE MADE UP FROM WHOLE SYNTACTIC CONSTRUCTIONS RATHER THAN FROM SINGLE WORDS; A PROCEDURAL CONDITION, IN WHICH PROGRAMS WERE WRITTEN WORD BY WORD AS USUAL, BUT A WELL-DEFINED PROCEDURE WAS PRESCRIBED TO HELP SUBJECTS WRITE NESTED CONDITIONALS CORRECTLY; AND A PLAIN CONDITION RESEMBLING STANDARD PROGRAMMING TUITION, IN WHICH SUBJECTS WERE TOLD THE STRUCTURE OF THE LANGUAGE BUT WERE GIVEN NO GUIDE TO HELP IN WRITING. SIGNIFICANTLY MORE ERROR-FREE PROGRAMS WERE WRITTEN IN THE PROCEDURAL CONDITION THAN IN THE PLAIN CONDITION, SHOWING THAT EXPLICIT PROCEDURES CAN IMPROVE PROGRAMMING SUCCESS, AT LEAST IN THESE CONDITIONS. IN THE AUTOMATIC CONDITION THE SUCCESS RATE WAS STILL HIGHER, SHOWING THAT THE PROCEDURE WE USED COULD STILL BE IMPROVED. THESE RESULTS, AND THE OUTCOMES OF FURTHER ANALYSES, BEAR ON RECOMMENDATIONS BY THE "STRUCTURED PROGRAMMING" SCHOOL TO FOLLOW EXPLICIT PROCEDURES WHEN WRITING PROGRAMS, AND ALSO ON PREVIOUS WORK ON THE DESIGN OF EASILY-USED PROGRAMMING LANGUAGES. (A)

392 PROGRAMMING LANGUAGES

SIME, M.E., ARBLASTER, A.T., & GREEN, T.R.G. STRUCTURING THE PROGRAMMER'S TASK. JOURNAL OF OCCUPATIONAL PSYCHOLOGY, 1977, 50, 205-216.

DESCRIPTION:

COMPUTER PROGRAMMING COULD BE MADE EASIER. THIS PAPER GIVES A SHORT ACCOUNT OF THE AUTHORS' EMPIRICAL STUDIES OF PROGRAMMING, SHOWING THAT VERY CONSIDERABLE IMPROVEMENTS CAN BE OBTAINED BOTH FOR BEGINNERS AND FOR PROFESSIONALS. THE IMPROVEMENTS ARE RELATED TO, THOUGH NOT WHOLLY DEPENDENT ON, THE NEW CONCEPTS OF 'STRUCTURED PROGRAMMING'; WE HAVE FOUND WAYS TO IMPROVE BOTH THE PROGRAMMING LANGUAGE ITSELF AND THE PROCEDURES USED BY THE PROGRAMMER. PSYCHOLOGICAL EXPLANATIONS OF THE EFFECTS ARE OFFERED AND THE LIMITATONS OF PRESENT FINDINGS ARE NOTED. (A)

12P, 18R.

393 PROGRAMMING LANGUAGES

SIME, M.E., FITTER, M., & GREEN, T. WHY IS PROGRAMMING COMPUTERS SO HARD? NEW BEHAVIOUR, SEPTEMBER 1975, 378-381.

DESCRIPTION:

THE PROBABILITY OF WRITING A CORRECTLY UNDERSTOOD PROGRAM CAN BE CONSIDERABLY IMPROVED BY BETTER PROGRAMMING LANGUAGE DESIGN. PROGRAMMING LANGUAGES ARE GENERALLY DESIGNED, HOWEVER, IN THE TOTAL ABSENCE OF EMPIRICAL DATA AND PRIMARILY ON THE BASIS OF INTUITION AND GUESSWORK. THIS PAPER BRIEFLY OUTLINES THE ROLE THAT PSYCHOLOGY SHOULD PLAY IN DESIGNING MORE EFFECTIVE PROGRAMMING LANGUAGES. (MEA)

4P, 0R.

394 PROGRAMMING LANGUAGES

SIME, M.E., GREEN, T.R.G., & GUEST, D.J. PSYCHOLOGICAL EVALUATION OF TWO CONDITIONAL CONSTRUCTIONS USED IN COMPUTER LANGUAGES. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1973, 5, 105-113.

DESCRIPTION:

THERE IS A NEED FOR EMPIRICAL EVALUATION OF PROGRAMMING LANGUAGES FOR UNSKILLED USERS, BUT IT IS MORE EFFECTIVE TO COMPARE SPECIFIC FEATURES COMMON TO MANY LANGUAGES THAN TO COMPARE COMPLETE LANGUAGES. THIS CAN BE DONE BY DEVISING MICRO-LANGUAGES STRESSING THE FEATURE OF INTEREST, TOGETHER WITH A SUITABLE SUBJECT MATTER FOR THE PROGRAMS. TO ILLUSTRATE THE POWER OF THIS APPROACH TWO CONDITIONAL CONSTRUCTIONS ARE COMPARED: A NESTABLE CONSTRUCTION, LIKE THAT OF ALGOL 60, AND A BRANCH-TO-LABEL CONSTRUCTION, AS USED IN MANY SIMPLER LANGUAGES. THE FORMER IS EASIER FOR UNSKILLED SUBJECTS. POSSIBLE REASONS FOR THIS FINDING ARE DISCUSSED. (A)

9P, 8R.

395 PROGRAMMING LANGUAGES

SIME, M.E., GREEN, T.R.G., & GUEST, D.J. SCOPE MARKING IN COMPUTER CONDITIONALS -- A PSYCHOLOGICAL EVALUATION. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1977, 9, 107-118.

DESCRIPTION:

IN A PREVIOUS PAPER THE AUTHORS REPORTED THAT IT WAS EASIER FOR NON-PROGRAMMERS TO LEARN TO USE NESTED CONDITIONAL CONSTRUCTIONS THAN JUMPING, OR BRANCH-TO-LABEL, CONSTRUCTIONS; HOWEVER, AS ONLY SINGLE SITUATIONS WERE STUDIED, THE CONCLUSIONS WERE NECESSARILY RESTRICTED. THE PRESENT STUDY EXTENDS THE COMPARISON TO THE MORE GENERAL CASE WHERE NESTING REQUIRES 'SCOPE MARKERS' TO DISAMBIGUATE THE SYNTAX. THE RESULTS SHOWED IF THE SCOPE MARKERS WERE SIMPLY THE BEGIN AND END OF ALGOL 60 (ABBREVIATED NEST-BE) THEN THE ADVANTAGE OF NESTING OVER JUMPING WAS WEAKENED; BUT IF THE SCOPE MARKERS CARRIED REDUNDANT INFORMATION ABOUT THE CONDITIONAL TESTED (NEST-INE) (FOR IF-NOT-END) PERFORMANCE WAS EXCELLENT, PARTICULARLY AT DEBUGGING. IT SEEMS NECESSARY TO DISTINGUISH SEQUENCE INFORMATION IN A PROGRAM, WHICH DESCRIBES THE ORDER IN WHICH THINGS ARE DONE, FROM TAXON INFORMATION, WHICH DESCRIBES THE CONDITIONS UNDER WHICH A GIVEN ACTION IS PERFORMED. CONVENTIONAL PROGRAMMING LANGUAGES OBSCURE THE TAXON INFORMATION. THE ADVANTAGE OF NESTING OVER JUMPING, WE SPECULATE, IS IN CLARIFYING THE SEQUENCE INFORMATION BY REDUNDANT RE-CODING IN SPATIAL TERMS; THE ADDED ADVANTAGE OF NEST-INE OVER NEST-BE IS THAT IT CLARIFIES THE TAXON INFORMATION. IT IS BECAUSE DEBUGGING REQUIRES TAXON INFORMATION THAT NEST-INE IS SO MUCH SUPERIOR. ON THIS VIEW ONE WOULD EXPECT THAT IN DECISION TABLE AND PRODUCTION SYSTEM LANGUAGES, WHERE THE TAXON INFORMATION IS EXPLICIT BUT THE SEQUENCE INFORMATION IS OBSCURED, THE REVERSE PHENOMENA SHOULD OCCUR. BECAUSE DEBUGGING REQUIRES SEQUENCE INFORMATION AS WELL AS TAXON INFORMATION, A DEVICE THAT CLARIFIED THE SEQUENCE WOULD GREATLY IMPROVE SUCH LANGUAGES. (A)

12P, 11R.

396 SOFTWARE DEVELOPMENT

SLAUGHTER, J.B. UNDERSTANDING THE SOFTWARE PROBLEM. AFIPS CONFERENCE PROCEEDINGS, 1974, 43, 333-336.

397 COMPARISON OF NORMAL BATCH PROCESSING WITH FAST TURNAROUND

SMITH, L.B. A COMPARISON OF BATCH PROCESSING AND INSTANT TURNAROUND. COMMUNICATIONS OF THE ACM, 1967, 10, 495-500.

DESCRIPTION:

A STUDY OF THE PROGRAMMING EFFORTS OF STUDENTS IN AN INTRODUCTORY PROGRAMMING COURSE IS PRESENTED AND THE EFFECTS OF HAVING INSTANT TURNAROUND (A FEW MINUTES) AS OPPOSED TO CONVENTIONAL BATCH PROCESSING WITH TURNAROUND TIMES OF A FEW HOURS ARE EXAMINED. AMONG THE ITEMS COMPARED ARE THE NUMBER OF COMPUTER RUNS PER TRIP TO THE COMPUTATION CENTER, PROGRAM PREPARATION TIME, KEYPUNCHING TIME, DEBUGGING TIME, NUMBER OF RUNS, AND ELAPSED TIME FROM THE FIRST RUN TO THE LAST RUN ON EACH PROBLEM. EVEN THOUGH THE RESULTS ARE INFLUENCED BY THE FACT THAT "BONUS POINTS" WERE GIVEN FOR COMPLETION OF A PROGRAMMING PROBLEM IN LESS THAN A SPECIFIED NUMBER OF RUNS, THERE IS EVIDENCE TO SUPPORT "INSTANT" OVER "BATCH". (A)

6P, 4R.

398 STRUCTURED PROGRAM VALIDATION

SNOWDEN, R.A. PEARL: AN INTERACTIVE SYSTEM FOR THE PREPARATION AND VALIDATION OF STRUCTURED PROGRAMS. SIGPLAN NOTICES, MARCH 1972, 7(3), 9-26.

DESCRIPTION:

THE PEARL SYSTEM IS AN ATTEMPT TO PROVIDE AN ENVIRONMENT FOR THE WRITING OF CORRECT PROGRAMS. FACILITIES ARE PROVIDED FOR THE CONSTRUCTION AND FILING OF STRUCTURED PROGRAMS, WHILST TECHNIQUES HAVE BEEN DEVELOPED FOR THE INCLUSION OF ASSERTIONS INVOLVING ABSTRACT OPERATIONS AND DATA TYPES. AS A RESULT, PROGRAMS, POSSIBLY INCOMPLETE, CAN BE COMPILED AND EXECUTED, ANY ERROR COMMUNICATION WITH THE PROGRAMMER BEING IN TERMS OF THE APPROPRIATE LEVEL OF HIS SOURCE PROGRAM. (A)

399 SOFTWARE DESIGN

SPITZEN, J.M., LEVITT, K.N., & ROBINSON, L. AN EXAMPLE OF HIERARCHICAL DESIGN AND PROOF (TECHNICAL REPORT NO. SRI-4079-TR-2). MENLO PARK, CALIFORNIA: STANFORD RESEARCH INSTITUTE, MARCH 1976. (NTIS NO. AD A021574)

400 PROGRAMMING

STANDISH, T.A. OBSERVATIONS AND HYPOTHESES ABOUT PROGRAM SYNTHESIS MECHANISMS (AUTOMATIC PROGRAMMING MEMO 9, REPORT NO. 2780). CAMBRIDGE, MASSACHUSETTS: BOLT BERANEK AND NEWMAN, COMPUTER SCIENCE DIVISION, DECEMBER 1973.

401 SOFTWARE DESIGN

STAY, J.F. HIPO AND INTEGRATED PROGRAM DESIGN. IBM SYSTEMS JOURNAL, 1976, 2, 143-154.

DESCRIPTION:

DISCUSSED IS A PROCEDURE OF HIERARCHICAL FUNCTIONAL DESIGN BY WHICH PROGRAMMING PROJECTS CAN BE ANALYZED INTO SYSTEM, PROGRAM, AND MODULE LEVELS. IT IS SHOWN THAT PROGRAM DESIGN IS MADE MORE EFFICIENT BY APPLYING HIERARCHY PLUS INPUT-PROCESS-OUT (HIPO) TECHNIQUES AT EACH LEVEL TO FORM AN INTEGRATED VIEW OF ALL LEVELS. (A)

12P, 10R.

402 PROGRAMMING LANGUAGES
STEELE, G.L., JR. MACARONI IS BETTER THAN SPAGHETTI. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 60-66 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 60-66).
DESCRIPTION:
WE PRESENT A STACK IMPLEMENTATION OF MULTIPLE ENVIRONMENTS SIMILAR IN PRINCIPLE TO THAT OF BOBROW AND WEGBREIT, BUT BASED ON A MODEL WHICH PROVIDES BOTH STATIC AND DYNAMIC SCOPING. WE NOTE SOME OF THE PRAGMATIC CONSEQUENCES OF THIS CHOICE OF MODELS; ONE IS THAT NO UNNECESSARY CONTROL STACK IS RETAINED FOR CERTAIN IMPORTANT CONSTRUCTIONS SUCH AS "UPWARD FUNARGS" AND COROUTINES. WE ALSO DISCUSS THE CORRECT TREATMENT OF EXIT FUNCTIONS, AND THE NEED FOR "ENTRY FUNCTIONS" IF DYNAMIC SWITCHING OF CONTROL CONTEXTS IS TO BE CONSISTENT. (A)
7P, 16R.

403 TIME-SHARING
ST. GERMAIN, J.M. CONVERSATIONAL TIME SHARING -- THE PROGRAM DEVELOPMENT BASE OF THE 70'S (TECHNICAL REPORT NO. TR 00 2135). IBM CORP., 1970.
DESCRIPTION:
PROVIDES SEVERAL ARGUMENTS AND SOME DATA ON THE SUPERIORITY OF INTERACTIVE PROGRAMMING OVER BATCH PROGRAMMING. (O)

404 SOFTWARE ENGINEERING
STOCKENBERG, J.E., & VANDAM, A. SRUCT PROGRAMMING ANALYSIS SYSTEM. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 384-389.

405 COMPUTER PERSONNEL
STREETER, D.N. PRODUCTIVITY OF COMPUTER-DEPENDENT WORKERS. IBM SYSTEMS JOURNAL, 1975, 14, 292-305.
DESCRIPTION:
BEGINNING WITH A DESCRIPTION OF VARIOUS DEGREES OF COMPUTER DEPENDENCY AMONG WORKERS, A MODEL OF THE WORKER-COMPUTER PROCESS IS CONSTRUCTED. THE MODEL DEMONSTRATES THE CHARACTERISTIC FORMS OF FUNCTIONAL DEPENDENCIES AND SUGGESTS WAYS IN WHICH THESE DEPENDENCIES CAN BE EVALUATED. KEY AMONG THE MANY CONSIDERATIONS DISCUSSED ARE SUCH PROCESS CHARACTERISTICS AS SYSTEM CONGESTION, NEEDS AND HABITS OF USERS, AND RELATIVE COSTS. (A)
14P, 9R.

406 SOFTWARE DESIGN
STREVELER, D.J. "DESIGNING BY COMMITTEE" WORKS -- SOMETIMES. DATAMATION, MARCH 1978, 24(3), 117; 119-120.
DESCRIPTION:
THE DESIGN OF COMPLEX SYSTEMS IS A HEURISTIC, RATHER THAN DETERMINISTIC, PROCEDURE. DESIGNING BY COMMITTEE AIDS DESIGN BY ALLOWING THE INTERCHANGE OF INSIGHTS, IDEAS, AND EXPERIENCES. ALTHOUGH IT IS NOT THE CHEAPEST METHOD OF DESIGN, IT IS THE FASTEST. THIS PAPER INFORMALLY DISCUSSES THE RELATIVE ADVANTAGES AND DISADVANTAGES OF DESIGNING BY COMMITTEE. (MEA)
3P, 0R.

407 PROGRAMMING
STRIZENEC, M. SOME APPROACHES TO AN ANALYSIS OF PROGRAMMER THINKING ACTIVITY. STUDIA PSYCHOLOGICA, 1974, 16, 64-66.

408 SOFTWARE DEVELOPMENT AIDS

STUCKI, L.G. AUTOMATED TOOLS AND TECHNIQUES ASSISTING IN SOFTWARE DEVELOPMENT. IN E. HOROWITZ (ED.). PRACTICAL STRATEGIES FOR DEVELOPING LARGE SOFTWARE SYSTEMS. READING, MASSACHUSETTS: ADDISON-WESLEY, 1975, 171-189.

DESCRIPTION:

THE PROBLEM OF SOFTWARE VALIDATION REQUIRES A THOROUGH UNDERSTANDING OF SOFTWARE BEHAVIOR. THE AUTOMATIC GENERATION OF SELF-METRIC SOFTWARE OFFERS A POWERFUL AND USEFUL APPROACH TO THE MEASUREMENT OF SOFTWARE BEHAVIOR AND THE VERIFICATION AND VALIDATION OF EVOLVING SYSTEMS. IN ORDER TO INCREASE SOFTWARE RELIABILITY, WE MUST DESIGN WITH VERIFICATION AND VALIDATION CONSTANTLY IN MIND AND CONTINUE TO DEVELOP AN INTEGRATED SET OF AUTOMATED SUPPORT TOOLS. (MEA)

19P, 6R.

409 SOFTWARE COMPLEXITY

SULLIVAN, J.E. ENGINEERING OF QUALITY SOFTWARE SYSTEMS (VOL.5): MEASURING THE COMPLEXITY OF COMPUTER SOFTWARE (REPORT NO. MTR-2648-VOL-5). BEDFORD, MASSACHUSETTS: MITRE CORP., JANUARY 1975. (NTIS NO. AD A007770)

DESCRIPTION:

THIS REPORT PRESENTS SEVERAL MEASURES OF COMPUTER PROGRAM COMPLEXITY, IN THE SENSE OF COMPREHENSIBILITY OR INTELLECTUAL MANAGABILITY. THE MEASURES CONSIDER THE PROGRAM AS AN ABSTRACT PROCESS, AND SO ARE INDEPENDENT OF PROGRAMMING LANGUAGE OR IMPLEMENTATION DETAILS. (A)

410 MANAGING THE SOFTWARE DEVELOPMENT PROCESS

SUNG, D. ENGINEERING PROCESSES IN MANAGING SOFTWARE SYSTEM DEVELOPMENT. COMPUTER PERSONNEL, 1977, 7(3), 7-13.

411 PROGRAMMING PRACTICES

SWANSON, E.B. COMPUTER APPLICATION SYSTEM DEVELOPMENT: SOME IMPLICATIONS FOR PROGRAMMING PRACTICE. DATA MANAGEMENT, MAY 1976, 34-38.

412 REQUIREMENTS LANGUAGES

TEICHROEW, D. A SURVEY OF LANGUAGES FOR STATING REQUIREMENTS FOR COMPUTER-BASED INFORMATION SYSTEMS. AFIPS CONFERENCE PROCEEDINGS, 1972, 41, 1203-1224.

413 COMPUTER AIDS FOR PROGRAMMING

TEITELMAN, W. PILOT: A STEP TOWARD MAN-COMPUTER SYMBIOSIS (TECHNICAL REPORT NO. MAC-TR-32). CAMBRIDGE, MASSACHUSETTS: MASSACHUSETTS INSTITUTE OF TECHNOLOGY, SEPTEMBER 1966. (NTIS NO. AD 638446)

414 AUTOMATIC PROGRAMMING

TEITELMAN, W. AUTOMATED PROGRAMMING -- THE PROGRAMMER'S ASSISTANT. AFIPS CONFERENCE PROCEEDINGS, 1972, 41, 917-921.

415 COMPUTER AIDS FOR PROGRAMMING

TEITELMAN, W. "DO WHAT I MEAN": THE PROGRAMMER'S ASSISTANT. COMPUTERS AND AUTOMATION, APRIL 1972, 21(4), 8-11.

DESCRIPTION:

THIS ARTICLE DEALS WITH THE DESIGN AND ACTUAL IMPLEMENTATION IN A COMPUTER PROGRAMMING SYSTEM OF "A PROGRAMMER'S ASSISTANT". THE GENERAL FUNCTION OF THE "PROGRAMMER'S ASSISTANT" IS TO MAKE IT POSSIBLE FOR THE HUMAN PROGRAMMER TO SAY TO THE COMPUTER "DO WHAT I MEAN" INSTEAD OF "DO WHAT I SAY," AND "UNDO WHAT I JUST TRIED -- IT DID NOT WORK," INSTEAD OF LEAVING THE PROGRAMMER WITH THE SAD CONSEQUENCES OF HIS ACTUAL INSTRUCTIONS.

IN OTHER WORDS, THE PROGRAMMER'S ASSISTANT DEALS WITH SUCH FACTORS AS: EASE OF INTERACTION, LEVEL OF INTERACTION, FORGIVENESS FOR ERRORS (BOTH SPELLING ERRORS AND ERRORS OF THOUGHT), GOING BACK AND TAKING A DIFFERENT PATH, CHANGING ONE'S MIND, ETC., AND IN GENERAL, THE PROGRAMMER'S ENVIRONMENT.

THIS AREA OF IMPROVEMENT IN INTERACTIVE PROGRAMMING IS IMPORTANT. FOR MANY APPLICATIONS, THE PROGRAMMER'S ENVIRONMENT INFLUENCES, AND TO A LARGE EXTENT DETERMINES, WHAT SORT OF PROBLEM HE CAN TACKLE, AND HOW FAR HE CAN GO IN A GIVEN TIME. IF THE "ENVIRONMENT" IS "COOPERATIVE" AND "HELPFUL," THEN THE PROGRAMMER CAN BE MORE AMBITIOUS AND PRODUCTIVE. IF NOT, HE MAY SPEND MUCH OF HIS TIME AND ENERGY PERFORMING ROUTINE CLERICAL TASKS AND "FIGHTING THE SYSTEM." (A)

4P, 3R.

416 COMPUTER AIDS FOR PROGRAMMING

TEITELMAN, W. TOWARD A PROGRAMMING LABORATORY. IN P. NAUR, B. RANDALL, & J. N. BUXTON (EDS.), SOFTWARE ENGINEERING: CONCEPTS AND TECHNIQUES. NEW YORK: PETROCELLI/CHARTER, 1976, 275-287.

DESCRIPTION:

THIS PAPER DISCUSSES THE FEASIBILITY AND DESIRABILITY OF CONSTRUCTING A "PROGRAMMING LABORATORY" WHICH WOULD CO-OPERATE WITH THE USER IN THE DEVELOPMENT OF HIS PROGRAMS, FREEING HIM TO CONCENTRATE MORE FULLY ON THE CONCEPTUAL DIFFICULTIES OF THE PROBLEM HE WISHES TO SOLVE. EXPERIENCE WITH SIMILAR SYSTEMS IN OTHER FIELDS INDICATES THAT SUCH A SYSTEM WOULD SIGNIFICANTLY INCREASE THE PROGRAMMER'S PRODUCTIVITY.

THE PILOT SYSTEM, IMPLEMENTED WITHIN THE INTERACTIVE BBN LISP SYSTEM, IS A STEP IN THE DIRECTION OF A PROGRAMMING LABORATORY. PILOT OPERATES AS AN INTERFACE BETWEEN THE USER AND HIS PROGRAMS, MONITORING BOTH THE REQUESTS OF THE USER AND THE OPERATION OF HIS PROGRAMS. FOR EXAMPLE, IF PILOT DETECTS AN ERROR DURING THE EXECUTION OF A PROGRAM, IT TAKES THE APPROPRIATE CORRECTIVE ACTION BASED ON PREVIOUS INSTRUCTIONS FROM THE USER. SIMILARLY, THE USER CAN GIVE DIRECTIONS TO PILOT ABOUT THE OPERATION OF HIS PROGRAMS, EVEN WHILE THEY ARE RUNNING, AND PILOT WILL PERFORM THE WORK REQUIRED. IN ADDITION, THE USER CAN EASILY MODIFY PILOT BY INSTRUCTING IT ABOUT ITS OWN OPERATION AND THUS DEVELOP HIS OWN LANGUAGE AND CONVENTIONS FOR INTERACTING WITH PILOT.

SEVERAL EXAMPLES ARE PRESENTED. (A)

15P, 10R.

417 STRUCTURED PROGRAMMING

TENNY, T. STRUCTURED PROGRAMMING IN FORTRAN. DATAMATION, JULY 1974, 20, PP. 110-111; 113; 115.

DESCRIPTION:

THE INDUSTRY'S INVESTMENT IN FORTRAN WILL KEEP IT AROUND A WHILE. HOWEVER, STRUCTURED PROGRAMMING CAN BE DONE IN FORTRAN IF CERTAIN RULES ARE FOLLOWED. NUMBERING OF STATEMENTS SHOULD ONLY BE IN INCREASING ORDER. BLOCKING AND NESTING SHOULD BE INDICATED BY BLANK COMMENTS AND INDENTATION. IF, GOTO, AND COMPUTED GOTO STATEMENTS SHOULD BE USED FOR THE PURPOSES OF REPEAT, WHILE, CASE, AND MORE GENERALIZED IF-THEN (-ELSE) CONSTRUCTS. THESE SHOULD ALSO BE DESIGNATED BY STANDARD COMMENTS. THE INTENT OF STRUCTURED PROGRAMMING, MODULARITY, TOP-DOWN DESIGN, CAN BE CARRIED OUT IN FORTRAN WITH PROPER MANAGEMENT AND PROFESSIONAL STANDARDS. (GDC)

4P, 6R.

418 GENERAL DISCUSSION OF HUMAN FACTORS IN COMPUTER SYSTEMS

TESTA, C.J. BEHAVIORAL FACTORS IN INFORMATION SYSTEMS. COMPUTERS AND PEOPLE, APRIL 1974, 23(4), 13-17.

DESCRIPTION:

THE NEED FOR BETTER UNDERSTANDING OF HUMAN BEHAVIOR IN INFORMATION SYSTEMS IS BECOMING INCREASINGLY APPARENT. TRADITIONALLY, INFORMATION SPECIALISTS HAVE CONCENTRATED THEIR EFFORTS ON HARDWARE/SOFTWARE PROBLEMS. AS A RESULT, SOPHISTICATED INFORMATION SYSTEMS WERE OFTEN DEVELOPED, BUT PEOPLE EXPERIENCED DIFFICULTY IN INTERACTING WITH THE COMPLEX SYSTEMS. SINCE INFORMATION SYSTEMS ARE USED, OPERATED, AND MAINTAINED BY PEOPLE, THE DESIGN OF EFFECTIVE INFORMATION SYSTEMS WILL ONLY RESULT IF MEN'S BEHAVIORAL CAPABILITIES ARE TAKEN INTO CONSIDERATION. IN THIS ARTICLE, MAN'S PERCEPTUAL AND COGNITIVE CAPABILITIES WILL BE EXAMINED AS IMPORTANT DETERMINANTS OF THE DESIGN OF INFORMATION SYSTEMS. (A, ABBR)

5P, 12R.

419 PROGRAMMING

THAYER, T.A. UNDERSTANDING SOFTWARE THROUGH ANALYSIS OF EMPIRICAL DATA (TECHNICAL REPORT NO. R-77-207). REDONDO BEACH, CALIFORNIA: TRW, UNDATED.

DESCRIPTION:

THIS PAPER DISCUSSES THE COLLECTION AND ANALYSIS OF DATA AVAILABLE DURING DEVELOPMENT, TESTING, AND OPERATIONAL USE OF SOTWARE SYSTEMS AS A MEANS OF DETERMINING SOFTWARE QUALITY IN QUANTIFIABLE TERMS. THE APPROACH TO DATA COLLECTION AND ANALYSIS TAKEN BY TRW IN A STUDY OF FOUR SOFTWARE SYSTEMS IS DESCRIBED INCLUDING SOME STUDY RESULTS AND IDENTIFICATION OF NECESSARY IMPROVEMENTS IN THE COLLECTION AND ANALYSIS PROCESSES. THIS PAPER TREATS BOTH THE LONG-RANGE AND THE NEAR-TERM PAYOFFS OF SUCH STUDIES IN AN ATTEMPT TO ANSWER THE QUESTION, "WHY COLLECT DATA AT ALL?" (A)

23P, 12R.

420 NATURAL-LANGUAGE DIALOGUE

THOMAS, J.C. A METHOD FOR STUDYING NATURAL LANGUAGE DIALOGUE (TECHNICAL REPORT RC-5882). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, FEBRUARY 1976. (NTIS NO. AD A041288)

421 QUANTIFIERS IN QUERY LANGUAGES

THOMAS, J.C. QUANTIFIERS AND QUESTION-ASKING (TECHNICAL REPORT NO. RC 5866). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, FEBRUARY 1976. (NTIS NO. AD A043032)

DESCRIPTION:

DATA CONCERNING THE USE OF UNIVERSAL QUANTIFIERS IN QUESTION-ASKING IS PRESENTED. THESE DATA WERE COLLECTED IN A VARIETY OF PROCEDURES USING NON-PROGRAMMERS. THESE NON-PROGRAMMERS VARIOUSLY TRANSLATED ENGLISH QUESTIONS INTO A QUERY LANGUAGE, GENERATED THEIR OWN ENGLISH QUESTIONS, TRANSLATED VENN DIAGRAMS INTO ENGLISH OR VICE VERSA, GAVE JUDGMENTS ABOUT THE CONSISTENCY OF TWO ENGLISH STATEMENTS, OR MANUALLY LOOKED UP ANSWERS TO QUESTIONS. SUBJECTS SHOWED CONSIDERABLE DIFFICULTY WITH THE LOGICIAN'S NOTATIONS OF SET RELATIONS (EXCEPT DISJUNCTION) ON ALL TASKS. THE INTERPRETATIONS GIVEN QUANTIFIED SENTENCES VARIED BETWEEN SUBJECTS ON A GIVEN TASK AND EVEN WITHIN A SUBJECT, BETWEEN TASKS. GENERALLY SPEAKING, SUBJECTS GAVE INTERPRETATIONS CONSISTENT WITH QUANTIFIED NATURAL LANGUAGE QUESTIONS OR VENN DIAGRAMS, BUT NOT EQUIVALENT TO THEM. SUBJECTS USED EXPLICIT SET SPECIFICATIONS RARELY IN SPONTANEOUS ENGLISH.

TENTATIVE SUGGESTIONS ARE MADE FOR THE DESIGN OF FORMAL AND NATURAL-LANGUAGE QUESTION-ANSWER INTERFACES. (A)

32P, 31R.

422 NATURAL-LANGUAGE DIALOGUE

THOMAS, J.C. A DESIGN-INTERPRETATION ANALYSIS OF NATURAL ENGLISH WITH APPLICATIONS TO MAN-COMPUTER INTERACTION (TECHNICAL REPORT RC-6581). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, JUNE 1977. (NTIS NO. AD A056121)

423 QUERY LANGUAGES

THOMAS, J.C., & GOULD, J.D. A PSYCHOLOGICAL STUDY OF QUERY BY EXAMPLE. AFIPS CONFERENCE PROCEEDINGS, 1975, 44, 439-445 (ALSO IBM REPORT RC-5124, IBM WATSON RESEARCH CENTER, YORKTOWN HEIGHTS, NEW YORK, NOVEMBER 1974).

DESCRIPTION:

THIRTY-NINE NON-PROGRAMMERS WERE TAUGHT ZLOOF'S QUERY BY EXAMPLE SYSTEM IN ORDER TO PROVIDE BEHAVIORAL DATA PRIOR TO IMPLEMENTATION. THIS TRAINING TOOK LESS THAN THREE HOURS. THEN SUBJECTS WERE GIVEN 40 TEST QUESTIONS IN ENGLISH WHICH THEY TRANSLATED INTO QUERY BY EXAMPLE. SUBJECTS ALSO RECORDED THE TIME TO WRITE EACH QUERY AND THEIR CONFIDENCE ABOUT BEING CORRECT. SIXTY-SEVEN PER CENT OF THE QUERIES WERE WRITTEN CORRECTLY. SUBJECTS AVERAGED 1.8 MINUTES TO WRITE QUERIES. QUERY DIFFICULTY COULD LARGELY BE PREDICTED FROM A LINEAR REGRESSION BASED ON OBJECTIVE COMPLEXITY MEASURES. CONFIDENCE RATING WAS ALSO AN EXCELLENT PREDICTOR OF QUERY DIFFICULTY. SUBJECTS HAD DIFFICULTY WITH QUANTIFICATION BUT LITTLE TROUBLE WITH LINKING VARIABLES, CONJUNCTIONS OR DISJUNCTIONS. IN A TWO-WEEK RETEST, FOUR OF SIX SUBJECTS SHOWED NEARLY PERFECT RETENTION OF THE SYSTEM RULES. RECOMMENDATIONS TO HELP PREVENT CERTAIN ERROR TYPES ARE MADE. (A)
7P, 17R.

424 SOFTWARE DESIGN

THOMAS, J.C., MALHOTRA, A., & CARROLL, J.M. AN EXPERIMENTAL INVESTIGATION OF THE DESIGN PROCESS (TECHNICAL REPORT RC-6702). YORKTOWN HEIGHTS, NEW YORK: IBM WATSON RESEARCH CENTER, AUGUST 1977.

425 PROGRAMMING LANGUAGES

TRAVIS, L., HONDA, M., LEBLANC, R., & ZEIGLER, S. DESIGN RATIONALE FOR TELOS, A PASCAL-BASED AI LANGUAGE. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 67-76. (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 67-76).

DESCRIPTION:

TELOS IS A PASCAL-BASED AI LANGUAGE INTENDED TO FACILITATE EFFICIENT DEVELOPMENT OF EFFICIENT, WELL-STRUCTURED PROGRAMS. THE DESIGN EMPHASIZES POWERFUL DATA ABSTRACTION AND CONTROL ABSTRACTION MECHANISMS RATHER THAN THE PROVISION OF PARTICULAR HIGH-LEVEL CONSTRUCTS. AMONG THE MANY CAPABILITIES OF TELOS ARE THOSE INTENDED TO MAKE IT ESPECIALLY SUITABLE FOR SYSTEMATIC AI MODEL BUILDING, FOR EXAMPLE, IN THE AREAS OF KNOWLEDGE REPRESENTATION, PLANNING, AND REASONING. AN EVENT FACILITY IS PROVIDED WHICH UNIFIES THE HANDLING OF CONDITIONAL INTERRUPTS (DEMONS), PROCESS SUSPENSION, PROCESS COMMUNICATION AND EXECUTION FAULTS. THE CONTEXT-DEPENDENT TELOS DATA BASE IS REFERENCEABLE EITHER ASSOCIATIVELY OR DIRECTLY. (A)
10P, 14R.

426 COMMAND LANGUAGE DESIGN BASED ON "MENTAL WORK" REQUIRED
TREU, S. INTERACTIVE COMMAND LANGUAGE DESIGN BASED ON REQUIRED MENTAL WORK. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1975, 7, 135-149.
DESCRIPTION:
ALTHOUGH THE DEFINITION OF "MENTAL WORK" REMAINS ELUSIVE, SYSTEMATIC MEANS/METHODS SHOULD BE CONSIDERED FOR GAINING EVIDENCE ABOUT INTERACTIVE LANGUAGE FEATURES REQUIRING MORE/LESS EFFORT OF THE HUMAN MIND. THE SUGGESTED APPROACH EMPLOYS A STRUCTURING OF THE USER'S CONCEPTUAL REFERENCE SPACES INTO SETS OF "ACTION PRIMITIVES", PECULIAR TO THE TYPE OF COMPUTER-AIDED TASK INVOLVED. AN INTERACTIVE COMMAND LANGUAGE CAN THEN BE REGARDED AS THE RANGE OF SOME TRANSFORMATION ON THE USER'S SET OF ACTION PRIMITIVES. THE NATURE AND EFFICIENCY OF THAT TRANSFORMATION, IN CONJUNCTION WITH THE INHERENT NUMBER OF MENTAL ASSOCIATION LINKS, ARE HYPOTHESIZED TO HAVE DIRECT RELATIONSHIPS TO THE LEVEL OF REQUIRED MENTAL WORK. THE USER'S DELAY OR "THINK TIME", EXPENDED IMMEDIATELY PRECEDING COMMAND UTILIZATION, IS ONE MEASURABLE QUANTITY THAT SHOULD BE USEFUL AS A WORK LEVEL INDICATOR. (A)
15P, 12R.

427 SYSTEM DOCUMENTATION
TSICHRITZIS, D. MODULAR SYSTEM DESCRIPTION (TECHNICAL REPORT NO. 33). TORONTO, ONTARIO, CANADA: UNIVERSITY OF TORONTO, DEPARTMENT OF COMPUTER SCIENCE, 1971.

428 PROGRAMMING LANGUAGES
TUCKER, A. VERY HIGH-LEVEL LANGUAGE DESIGN: A VIEWPOINT. COMPUTER LANGUAGES, 1975, 1, 3-16.
DESCRIPTION:
RECENT DEVELOPMENTS IN VERY HIGH-LEVEL LANGUAGE DESIGN INDICATE THAT THESE LANGUAGES HOLD GREAT PROMISE FOR IMPROVING THE LEVEL OF MAN-MACHINE COMMUNICATION, AND HENCE, IMPROVING COMPUTER AND PROGRAMMER UTILIZATION. (ESSENTIALLY, A VERY HIGH-LEVEL LANGUAGE ONE WHICH ALLOWS THE PROGRAMMER TO SPECIFY WHAT TO DO, RATHER THAN HOW TO DO IT.) THIS PAPER SURVEYS THESE DEVELOPMENTS, OUTLINES THE GOALS TO WHICH AN "IDEAL" VERY HIGH-LEVEL LANGUAGE SHOULD ASPIRE, AND THEN PRESENTS THE DESIGN OF A VERY HIGH-LEVEL LANGUAGE THAT WOULD MEET THESE GOALS. THIS DESIGN IS PRESENTED IN THE INTEREST OF LAYING BARE SOME BASIC DESIGN AND IMPLEMENTATION QUESTIONS THAT ARE INHERENT TO SUCH AN ACHIEVEMENT. THE PAPER THEN DISCUSSES THESE QUESTIONS, INDICATING BOTH OLD AND NEW RESEARCH PROBLEMS WHICH THEY SUGGEST. (A)
14P, 11R.

429 PROGRAM SYNTHESIS
ULRICH, J.W., & MOLL, R. PROGRAM SYNTHESIS BY ANALOGY. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 22-28 (ALSO SIGART NEWSLETTER, AUGUST 1977, NO. 64, 22-28).
DESCRIPTION:
BY EXTENDING A GIVEN ANALOGY, A KNOWN PROGRAM WHICH SOLVES A GIVEN PROBLEM IS CONVERTED TO A PROGRAM WHICH SOLVES A DIFFERENT BUT ANALOGOUS PROBLEM. THE DOMAINS OF THE TWO PROBLEMS NEED NOT BE THE SAME BUT THEY MUST BE RELATED BY AN INITIAL SPECIFIED ANALOGY. THERE ARE THREE FEATURES WHICH DISTINGUISH THE APPROACH. FIRST THE ANALOGY FORMATION EVOLVES GRADUALLY WITH THE SYNTHESIS OF THE NEW PROGRAM. SECONDLY, THE FORMATION OF THE ANALOGY IS DIRECTED BY THE CORRECTNESS PROOF OF THE KNOWN PROGRAM. FINALLY, THE OUTPUT OF THE SYNTHESIS PROCESS PRODUCES A CORRECTNESS PROOF FOR THE SYNTHESIZED PROGRAM. (A)
7P, 12R.

430 PROGRAM DESIGN LANGUAGE
VAN LEER, P. TOP-DOWN DEVELOPMENT USING A PROGRAM DESIGN LANGUAGE. IBM SYSTEMS JOURNAL, 1976, 2, 155-170.
DESCRIPTION:
DISCUSSED IS A PROGRAM DESIGN LANGUAGE -- A FORM OF PSEUDOCODE -- THAT HAS BEEN DEVELOPED AND USED TO ORGANIZE, TEACH, DOCUMENT, AND DEVELOP SOFTWARE SYSTEMS. AN EXAMPLE OF TOP-DOWN PROGRAM DESIGN ILLUSTRATES THE KEY STEPS IN USING THE LANGUAGE: DETERMINING THE REQUIREMENTS, ABSTRACTING THE FUNCTIONS, EXPANDING THE FUNCTIONS, AND VERIFYING THE FUNCTIONS. SYNTAX AND CONVENTIONS OF THE LANGUAGE ARE GIVEN IN AN APPENDIX. (O)
16P, 7R.

431 AUTOMATIC PROGRAMMING
WALDINGER, R.J., & LEE, R.C.T. PROW: A STEP TOWARD AUTOMATIC PROGRAM WRITING. PROCEEDINGS OF THE INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, 1969, 241-252.

432 PROGRAMMING LANGUAGES
WARREN, D.H.D., PEREIRA, L.M., & PEREIRA, F. PROLOG -- THE LANGUAGE AND ITS IMPLEMENTATION COMPARED WITH LISP. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 109-115 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 109-115).
DESCRIPTION:
PROLOG IS A SIMPLE BUT POWERFUL PROGRAMMING LANGUAGE FOUNDED ON SYMBOLIC LOGIC. THE BASIC COMPUTATIONAL MECHANISM IS A PATTERN MATCHING PROCESS ('UNIFICATION') OPERATING ON GENERAL RECORD STRUCTURES ('TERMS' OF LOGIC). WE BRIEFLY REVIEW THE LANGUAGE AND COMPARE IT ESPECIALLY WITH PURE LISP. THE REMAINDER OF THE PAPER DISCUSSES TECHNIQUES FOR IMPLEMENTING PROLOG EFFICIENTLY; IN PARTICULAR, WE DESCRIBE HOW TO COMPILE THE PATTERNS INVOLVED IN THE MATCHING PROCESS. THESE TECHNIQUES ARE AS INCORPORATED IN OUR DECSYSTEM-10 PROLOG COMPILER (WRITTEN IN PROLOG). THE CODE IT GENERATES IS COMPARABLE IN SPEED WITH THAT PRODUCED BY EXISTING DEC10 LISP COMPILERS. WE ARGUE THAT PATTERN MATCHING IS A BETTER METHOD FOR EXPRESSING OPERATIONS ON STRUCTURED DATA THAN CONVENTIONAL SELECTORS AND CONSTRUCTORS -- BOTH FOR THE USER AND FOR THE IMPLEMENTOR. (A)
7P, 15R.

433 PROGRAMMING LANGUAGES
WASSERMAN, A.I. ISSUES IN PROGRAMMING LANGUAGE DESIGN: AN OVERVIEW. SIGPLAN NOTICES, JULY 1975, 10(7), 10-12.
DESCRIPTION:
SEVERAL KEY QUESTIONS CAN BE RAISED CONCERNING THE DESIGN OF PROGRAMMING LANGUAGES. HOW DO WE DEVELOP A PROGRAMMING MECHANISM WHICH CAN ACCURATELY MIRROR LOGICAL THINKING? FURTHERMORE, HOW DO WE DEVELOP A TOOL WHICH IS SUITABLE FOR STEPWISE REFINEMENT OF THE PROBLEM FROM ITS ABSTRACT FORM TO ITS "ELABORATED" FORM IN A "NATURAL" WAY? LAST, HOW THEN DOES SUCH A LANGUAGE GET INTRODUCED AND ACCEPTED BY THE GENERAL PROGRAMMING COMMUNITY SO THAT IT RAISES THE QUALITY OF SOFTWARE PRODUCTION? THESE ARE THE MAIN QUESTIONS WHICH UNDERLIE PRESENTED RESEARCH AND DEVELOPMENT IN THE FIELD OF PROGRAMMING LANGUAGES. (A, ABBR.)
3P, 44R.

434 PROGRAMMING METHODOLOGY
WEGBREIT, B. GOAL-DIRECTED PROGRAM TRANSFORMATION. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1976, SE-2, 69-79.

435 HISTORY OF PROGRAMMING LANGUAGES
WEGNER, P. PROGRAMMING LANGUAGES -- THE FIRST 25 YEARS. IEEE TRANSACTIONS ON COMPUTERS, 1976, C-25, 1207-1225.

436 PROGRAMMING
WEINBERG, G.M. THE PSYCHOLOGY OF COMPUTER PROGRAMMING. NEW YORK: VAN NOSTRAND REINHOLD, 1971.
DESCRIPTION:
THE PRINCIPAL GOAL OF THIS BOOK IS TO ESTABLISH A FRAMEWORK FOR VIEWING COMPUTER PROGRAMMING AS A HUMAN ACTIVITY. THE AREAS CONSIDERED INCLUDE PROGRAMMING AS HUMAN PERFORMANCE, PROGRAMMING AS A SOCIAL ACTIVITY, PROGRAMMING AS AN INDIVIDUAL ACTIVITY, AND PROGRAMMING TOOLS. EXPERIMENTAL RESULTS ARE PRESENTED AND EACH CHAPTER IS FOLLOWED BY AN ANNOTATED BIBLIOGRAPHY. (MEA)
303P, 107R.

437 PROGRAMMER PERFORMANCE
WEINBERG, G.M. THE PSYCHOLOGY OF IMPROVED PROGRAMMING PERFORMANCE. DATAMATION, NOVEMBER 1972, 18(11), PP. 82-83; 85.
DESCRIPTION:
THIS PAPER DESCRIBES EXPERIMENTS THAT TEST THE IMPACT OF SPECIFIED GOALS ON PROGRAMMING PERFORMANCE. EXPLICITLY STATED GOALS AFFECT BOTH THE PROGRAMS THAT ARE PRODUCED AND THE ESTIMATED TIME REQUIRED FOR COMPLETION. THE CONFLICTING AND COMPLEMENTARY NATURE OF THE RELATIONS AMONG VARIOUS GOALS IS ALSO CONSIDERED. (MEA)
3P, 3R.

438 ERROR VERIFICATION
WEINBERG, G.M., & GRESSETT, G.L. AN EXPERIMENT IN AUTOMATIC VERIFICATION OF PROGRAMS. COMMUNICATIONS OF THE ACM, 1963, 6, 610-613.
DESCRIPTION:
VERIFICATION OF ERRORS INTRODUCED THROUGH THE STAGES OF KEYPUNCHING AND TRANSCRIPTION IS DISCUSSED. ALTHOUGH THE ARTICLE SPEAKS PRIMARILY TO KEYPUNCH AND SYNTAX ERRORS, IT MAKES SOME CONJECTURES ON HOW ERRORS IN CONTROL AND COMPUTATION STRUCTURES COULD BE DETECTED. THE PAPER DOES NOT DEAL WITH PROGRAM VERIFICATION ABOVE THE SYNTAX LEVEL AND IT IS NOT CONCERNED WITH "PROOFS" OF CORRECTNESS, FLOW ANALYSIS, INPUT OUTPUT, AND OTHER TOPICS AT THIS LEVEL.
4P, 2R.

439 PROGRAMMER PERFORMANCE

WEINBERG, G.M., & SCHULMAN, E.L. GOALS AND PERFORMANCE IN COMPUTER PROGRAMMING. HUMAN FACTORS, 1974, 16, 70-77.

DESCRIPTION:

IN ALL STUDIES OF HUMAN PERFORMANCE, THE EXPERIMENTER MUST BE CERTAIN THAT THE SUBJECT IS PERFORMING THE TASK THAT THE EXPERIMENTER BELIEVES HE HAS SET. OTHERWISE RESULTS BECOME UNINTERPRETABLE. EARLY STUDIES OF COMPUTER PROGRAMMING HAVE SHOWN SUCH WIDE VARIATIONS IN INDIVIDUAL PERFORMANCE THAT ONE MIGHT SUSPECT THAT SUBJECTS DIFFERED IN THEIR INTERPRETATION OF THE TASK. EXPERIMENTS ARE REPORTED WHICH SHOW HOW PROGRAMMING PERFORMANCE CAN BE STRONGLY INFLUENCED BY SLIGHT DIFFERENCES IN PERFORMING OBJECTIVES. CONCLUSIONS ARE DRAWN FROM THESE RESULTS REGARDING BOTH FUTURE EXPERIMENTATION AND MANAGEMENT PRACTICES IN COMPUTER PROGRAMMING. (A)

SUBJECTS INSTRUCTED TO RAPIDLY PRODUCE A PROGRAM TOOK FEWER RUNS AND PRODUCED MORE CORRECT, MORE MODIFIABLE, LESS EFFICIENT PROGRAMS THAN DID SUBJECTS INSTRUCTED TO PRODUCE EFFICIENT PROGRAMS. THE FORMER SUBJECTS ALSO GAVE MUCH MORE CONSERVATIVE PRIOR ESTIMATES OF EXPECTED COMPLETION TIME AND NUMBER OF RUNS. IN A SECOND STUDY, GROUPS OF PROGRAMMERS GIVEN THE SAME PROGRAMMING TASK WITH DIFFERENT OBJECTIVES (MINIMUM CORE, MINIMUM EXECUTION TIME, OUTPUT READABILITY, PROGRAM READABILITY, MINIMUM STATEMENTS, MINIMUM PROGRAMMING TIME) SUCCEEDED IN SATISFYING THOSE OBJECTIVES AT THE COST OF OTHERS. PROGRAMS WITH READABILITY OBJECTIVES WERE MOST READABLE, WHILE PROGRAMS WITH EXECUTION EFFICIENCY OBJECTIVES WERE JUDGED TO PRODUCE UNACCEPTABLE, UNREADABLE OUTPUTS. (HRR)

8P, 6R.

440 SOFTWARE DEVELOPMENT

WEISS, D.M. THE MUDD REPORT: A CASE STUDY OF NAVY SOFTWARE DEVELOPMENT PRACTICES (NRL REPORT 7909). WASHINGTON, D.C.: NAVAL RESEARCH LABORATORY, MAY 1975.

DESCRIPTION:

THE MUDD REPORT IS A STUDY OF NAVY SOFTWARE-DEVELOPMENT PRACTICES WHICH IS BASED ON A SERIES OF INTERVIEWS WITH THOSE RESPONSIBLE FOR THE DEVELOPMENT OF NAVY SYSTEMS. THE STUDY CHRONICLES THE DEVELOPMENT OF A FICTIONAL SYSTEM WITH REQUIREMENTS TYPICAL OF NAVY TACTICAL SYSTEMS CURRENTLY OPERATIONAL OR UNDER DEVELOPMENT. A HISTORY OF THE DECISIONS MADE DURING THE DEVELOPMENT OF THE SYSTEM IS FIRST GIVEN. FOLLOWING THE HISTORY IS AN ANALYSIS OF THE IMPACT OF EACH DECISION ON THE SOFTWARE DEVELOPED FOR THE SYSTEM AND ON THE LIFE-CYCLE OF THE SOFTWARE. FINALLY, A SET OF RECOMMENDATIONS FOR AVOIDING THE PITFALLS DESCRIBED IN THE REPORT IS GIVEN. THE RECOMMENDATIONS ARE DESIGNED TO ASSIST NAVY PROGRAM MANAGERS RESPONSIBLE FOR SOFTWARE DEVELOPMENT.

32P, 0R.

441 PROGRAMMING

WEISSMAN, L. PSYCHOLOGICAL COMPLEXITY IN COMPUTER PROGRAMS: AN INITIAL EXPERIMENT (TECHNICAL REPORT NO. CSRG-26). TORONTO, CANADA: UNIVERSITY OF TORONTO, COMPUTER SCIENCE RESEARCH GROUP, 1973.

442 PROGRAMMING

WEISSMAN, L. PSYCHOLOGICAL COMPLEXITY OF COMPUTER PROGRAMS: AN EXPERIMENTAL METHODOLOGY. SIGPLAN NOTICES, JUNE 1974, 9(6), 25-36.

DESCRIPTION:

IN ORDER TO REDUCE THE COMPLEXITY OF PROGRAMS, MANY IDEAS AND TECHNIQUES HAVE BEEN EXPOUNDED. HOWEVER, NO QUANTITATIVE EVIDENCE HAS BEEN GIVEN THAT THE QUALITY OF THE PROGRAMS HAS INDEED BEEN IMPROVED. WE BELIEVE THAT EXPERIMENTAL STUDIES SHOULD BE PERFORMED TO MEASURE THOSE FACTORS WHICH MAKE PROGRAMS DIFFICULT TO UNDERSTAND AND MAINTAIN. THE FIRST STEP IN SUCH RESEARCH IS TO ESTABLISH A SUITABLE EXPERIMENTAL METHODOLOGY. THIS PAPER DESCRIBES A SERIES OF EXPERIMENTS WHICH HAVE BEEN CONDUCTED WITH THE AIM OF ESTABLISHING SUCH A METHODOLOGY. (A)

12P, 20R.

443 PROGRAM COMPLEXITY

WEISSMAN, L.M. A METHODOLOGY FOR STUDYING THE PSYCHOLOGICAL COMPLEXITY OF COMPUTER PROGRAMS (TECHNICAL REPORT CSRG-37). TORONTO, ONTARIO, CANADA: UNIVERSITY OF TORONTO, COMPUTER SYSTEMS RESEARCH GROUP, AUGUST 1974.

DESCRIPTION:

THIS THESIS DEVELOPS A METHODOLOGY FOR EMPIRICALLY INVESTIGATING THE EFFECTS OF VARIOUS FACTORS ON THE PSYCHOLOGICAL COMPLEXITIES OF COMPUTER PROGRAMS. THE SPECIFIC FACTORS INVESTIGATED ARE: USE OF COMMENTS, CONTROL FLOW, PARAGRAPHING, CHOICE OF VARIABLE NAMES, AND LOCALITY OF DATA REFERENCES. TEN EXPERIMENTS WERE PERFORMED AND BOTH OBJECTIVE AND SUBJECTIVE MEASURES OF PERFORMANCE WERE COLLECTED. SIGNIFICANT RESULTS WERE OBTAINED FOR ALL FIVE FACTORS STUDIED. POSSIBLE DIRECTIONS FOR FUTURE RESEARCH ARE SUGGESTED. (MEA)

238P, 75R.

444 PROGRAMMING

WEISSMAN, L. EXPERIMENTAL METHODOLOGIES FOR STUDYING PROGRAMMING. PAPER PRESENTED AT THE 6TH CONGRESS OF THE INTERNATIONAL ERGONOMICS ASSOCIATION, UNIVERSITY OF MARYLAND, COLLEGE PARK, MARYLAND, 11-16 JULY 1976.

DESCRIPTION:

THE NEED FOR CLEAR, WELL-WRITTEN PROGRAMS IS NOW AN ACKNOWLEDGED FACT. CONSEQUENTLY, MANY TECHNIQUES, MOST NOTABLY THOSE UNDER THE BANNER OF STRUCTURED PROGRAMMING, HAVE BEEN PROPOSED FOR PRODUCING SUCH PROGRAMS. TOO FREQUENTLY, SUCH TECHNIQUES ARE PRESENTED WITHOUT ANY OBJECTIVE EVIDENCE THAT THEY ACTUALLY LEAD TO BETTER PROGRAMS. (A)

445 MAN-COMPUTER DIALOGUE

WEIZENBAUM, J. A COMPUTER PROGRAM FOR THE STUDY OF NATURAL LANGUAGE COMMUNICATION BETWEEN MAN AND MACHINE. CAMBRIDGE, MASSACHUSETTS: MASSACHUSETTS INSTITUTE OF TECHNOLOGY, DEPARTMENT OF ELECTRICAL ENGINEERING, SEPTEMBER 1965.

446 PROGRAMMING TOOLS AND STANDARDS

WHITTEN, D.E., & DEMAINE, P.A.D. A MACHINE AND CONFIGURATION INDEPENDENT FORTRAN: PORTABLE FORTRAN (PFORTRAN). IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 111-124.

447 BEHAVIORAL MISCONCEPTIONS WHICH MAY LEAD TO INFERIOR SYSTEMS

WILCOX, R.H. BEHAVIORAL MISCONCEPTIONS FACING THE SOFTWARE ENGINEER. IN J.T. TOU (ED.), SOFTWARE ENGINEERING: COINS III (PROCEEDINGS OF THE THIRD SYMPOSIUM ON COMPUTER AND INFORMATION SCIENCES HELD IN MIAMI BEACH, FLORIDA, DECEMBER 1969) (VOL. 2). NEW YORK: ACADEMIC PRESS, 1971, 285-287.

DESCRIPTION:

THIS ARTICLE BRIEFLY DISCUSSES A FEW MISCONCEPTIONS OF USER BEHAVIOR WHICH MAY LEAD AN INFORMATION SYSTEM DESIGNER TO DELIVER AN INAPPROPRIATELY DESIGNED SYSTEM. (HRR)

3P, OR.

448 AUTOMATIC PROGRAMMING

WILE, D., BALZER, R., & GOLDMAN, N. AUTOMATED DERIVATION OF PROGRAM CONTROL STRUCTURE FROM NATURAL LANGUAGE PROGRAM DESCRIPTIONS. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 77-84 (ALSO: SIGART NEWSLETTER, AUGUST 1977, NO. 64, 77-84).

DESCRIPTION:

THIS PAPER DESCRIBES A SYSTEM WHICH ORGANIZES A NATURAL LANGUAGE DESCRIPTION OF A PROGRAM INTO A CONVENTIONAL PROGRAM CONTROL STRUCTURE, AS A PART OF A LARGER SYSTEM FOR CONVERTING INFORMAL NATURAL LANGUAGE PROGRAM SPECIFICATIONS INTO RUNNING PROGRAMS. ANALYSIS OF THE INPUT PROGRAM FRAGMENTS USING A MODEL OF A HUMAN "READER" OF SPECIFICATIONS HAS BEEN FOUND TO BE A VERY SUCCESSFUL ADJUNCT TO CONVENTIONAL "PLANNING" METHODOLOGIES.

NATURAL LANGUAGE DESCRIPTIONS OF PROGRAMS CAN FREQUENTLY BE CHARACTERIZED AS "RUBBLE" -- A VERY LOOSELY ORGANIZED SET OF ALMOST INDEPENDENT DESCRIPTION FRAGMENTS. SUCH SPECIFICATIONS ARE OFTEN QUITE ROBUST, DUE TO A LARGE DEGREE OF REDUNDANCY; THEY ARE ALSO FREQUENTLY QUITE CONCISE, DUE TO RELIANCE ON THE READERS' INNATE KNOWLEDGE AND THEIR KNOWLEDGE OF THE APPLICATION DOMAIN. THIS PAPER DISCUSSES A PARADIGM FOR STRUCTURING THE PORTION OF "RUBBLE" PROGRAM DESCRIPTIONS WHICH MAPS INTO CONVENTIONAL PROGRAMMING LANGUAGE CONTROL CONSTRUCTS AND DEFINITION FACILITIES.

IN ORDER TO FOCUS ON STRUCTURING NATURAL LANGUAGE, IT IS NECESSARY TO INDICATE WHERE THIS MAPPING FITS IN THE BROADER SCHEME OF "UNDERSTANDING" NATURAL LANGUAGE PROGRAM DESCRIPTIONS. THE RESEARCH DESCRIBED BELOW IS THE BASIS FOR THE DESIGN OF AN INTERMEDIATE STAGE IN THE OPERATION OF THE SAFE SYSTEM, A SYSTEM DESIGNED AND IMPLEMENTED AT ISI TO PRODUCE FORMAL, OPERATIONAL SPECIFICATIONS FOR PROGRAMS DESCRIBED IN NATURAL LANGUAGE. IN PARTICULAR, A (PARENTHESIZED) NATURAL LANGUAGE DESCRIPTION OF A PROGRAM IS GIVEN TO THE SYSTEM -- A DESCRIPTION WHICH RETAINS MOST SEMANTIC AMBIGUITIES OF NATURAL LANGUAGE, BUT WHICH AVOIDS ITS SYNTACTIC AMBIGUITIES. THE INPUT FIRST GOES THROUGH A "DOMAIN ACQUISITION" PHASE WHICH ACQUIRES DOMAIN KNOWLEDGE RELATING THE OBJECTS AND ACTIONS OF THE MODELLED WORLD. THE "PLANNING PHASE", DESCRIBED HEREIN, IS THEN USED TO STRUCTURE THE INPUT INTO A PROGRAM IN CONVENTIONAL TERMS. FINALLY, A PHASE CONCERNED WITH THE RESOLUTION OF FINE DETAILS -- ANAPHORIC REFERENCE, TYPE CONVERSION, AND SOME SEQUENTIAL STRUCTURE RESOLUTION -- IS USED TO PRODUCE THE FINAL PROGRAM. THE RESPECTIVE PHASES DEAL IN TURN WITH THE DATA AND OPERATION STRUCTURE, THE PROGRAM DEFINITION AND CONTROL STRUCTURE, AND THE PROGRAM VARIABLE AND PARAMETER STRUCTURE.

THE SAFE SYSTEM MAKES OPERATIONAL SPECIFICATIONS MORE PRECISE BY FILLING IN THOSE DETAILS THAT WERE SUPPRESSED FROM THE SPECIFICATION BECAUSE THEY WERE DEEMED INFERABLE BY AN "INTELLIGENT READER". THESE SPECIFICATIONS MUST BE OPERATIONAL, SPECIFYING INFORMALLY AND AT A HIGH LEVEL, HOW SOMETHING IS TO BE DONE, NOT MERELY WHAT MUST BE ACHIEVED. THIS REQUIREMENT ENABLES THE CORRESPONDING FORMAL PROGRAM TO BE CONSTRUCTED WITHOUT ANY DEEP PROBLEM SOLVING ACTIVITY BY RESOLVING THE AMBIGUITIES CONTAINED OF PROGRAM WELL-FORMEDNESS RULES AND THE CONSTRAINTS OF THE APPLICATION DOMAIN. WHEN AN AMBIGUITY CANNOT BE RESOLVED BY THE SYSTEM, IT ASKS THE USER WHICH INTERPRETATION IS INTENDED. (A)

8P, 7R.

449 SOFTWARE ENGINEERING
WILKES, M.V. SOFTWARE ENGINEERING AND STRUCTURED PROGRAMMING. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1976, SE-2, 274-276.
DESCRIPTION:
THIS PAPER DISCUSSES THE REQUIREMENTS OF PROGRAMMERS WORKING IN VARYING ENVIRONMENTS IN RELATION TO SOFTWARE ENGINEERING, STRUCTURED PROGRAMMING, AND PROGRAM VERIFICATIONS. (A)
3P, 1R.

450 PROGRAMMING LANGUAGES
WILLIAMS, M.H. A NOTE ON THE AMBIGUITY OF THE COMMON STATEMENT. SIGPLAN NOTICES, NOVEMBER 1975, 10(11), 38-40.
DESCRIPTION:
THE MAIN REASON THAT USE OF THE COMMON STATEMENT FREQUENTLY LEADS TO ERRORS IS THAT THIS STATEMENT IS USED FOR THREE DIFFERENT PURPOSES. THIS AMBIGUITY MAKES IT DIFFICULT TO UNDERSTAND WHAT PURPOSE WAS INTENDED BY A PROGRAMMER. IT IS SUGGESTED THAT THREE SEPARATE STATEMENTS BE CREATED TO HANDLE THESE THREE FUNCTIONS. (MEA)
3P, 3R.

451 AUTOMATIC PROGRAMMING
WILLIAMS, M.H. A QUESTION-ANSWERING SYSTEM FOR AUTOMATIC PROGRAM SYNTHESIS. SIGPLAN NOTICES, JULY 1976, 11 (7), 63-68.

452 SOFTWARE RELIABILITY
WILLIAMS, R.D. MANAGING THE DEVELOPMENT OF RELIABLE SOFTWARE. IN PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON RELIABLE SOFTWARE. SIGPLAN NOTICES, JUNE 1975, 10(6), 3-8.

453 COMPUTER GRAPHICS
WILLIAMS, R., & GIDDINGS, G.M. A PICTURE-BUILDING SYSTEM. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1976, SE-2, 62-66.

454 COMPUTER PERSONNEL TURNOVER
WILLOUGHBY, T.C. COMPUTING PERSONNEL TURNOVER: A REVIEW OF THE LITERATURE. COMPUTER PERSONNEL, 1977, 7(1-2), 11-13.

455 PROGRAMMING LANGUAGE

WIRTH, N. ON CERTAIN BASIC CONCEPTS OF PROGRAMMING LANGUAGES (TECHNICAL REPORT NO. CS 65). STANFORD, CALIFORNIA: STANFORD UNIVERSITY, COMPUTER SCIENCE DEPARTMENT, MAY 1967. (NTIS NO. PB 176767)

DESCRIPTION:

RECENT DEVELOPMENT OF PROGRAMMING LANGUAGES HAVE LED TO THE EMERGENCE OF LANGUAGES WHOSE GROWTH SHOWED CANCEROUS SYMPTOMS: THE PROLIFERATION OF NEW ELEMENTS DEFIED EVERY CONTROL EXERCISED BY THE DESIGNERS, AND THE NATURE OF THE NEW CELLS OFTEN PROVED TO BE INCOMPATIBLE WITH THE EXISTING BODY. IN ORDER THAT A LANGUAGE BE FREE FROM SUCH SYMPTOMS, IT IS NECESSARY THAT IT BE BUILT UPON BASIC CONCEPTS WHICH ARE SOUND AND MUTUALLY INDEPENDENT. THE RULES GOVERNING THE LANGUAGE MUST BE SIMPLE, GENERALLY APPLICABLE, AND CONSISTENT.

IN PRACTICE, IT TURNS OUT THAT THERE EXISTS AN OPTIMUM IN THE NUMBER OF BASIC CONCEPTS, BELOW WHICH NOT ONLY IMPLEMENTABILITY OF THESE CONCEPTS ON ACTUAL COMPUTERS, BUT ALSO THEIR APPEAL TO HUMAN INTUITION BECOMES QUESTIONABLE BECAUSE OF THEIR HIGH DEGREE OF GENERALIZATION. THE FOLLOWING INFORMAL NOTES DO NOT ABOUND WITH READY-MADE SOLUTIONS, BUT IT IS HOPED THEY SHED SOME LIGHT ON SEVERAL RELATED SUBJECTS AND INHERENT DIFFICULTIES. THEY ARE INTENDED TO SUMMARIZE AND INTERRELATE VARIOUS IDEAS WHICH ARE PARTLY PRESENT IN EXISTING LANGUAGES AND PARTLY NEW. (A, ABBR.)
33P, 6R.

456 STEPWISE REFINEMENT

WIRTH, N., PROGRAM DEVELOPMENT BY STEPWISE REFINEMENT. COMMUNICATIONS OF THE ACM, 1971, 14, 221-227.

DESCRIPTION:

THE CREATIVE ACTIVITY OF PROGRAMMING -- TO BE DISTINGUISHED FROM CODING -- IS USUALLY TAUGHT BY EXAMPLES SERVING TO EXHIBIT CERTAIN TECHNIQUES. IT IS HERE CONSIDERED AS A SEQUENCE OF DESIGN DECISIONS CONCERNING THE DECOMPOSITION OF TASKS INTO SUBTASKS AND OF DATA INTO DATA STRUCTURES. THE PROCESS OF SUCCESSIVE REFINEMENT OF SPECIFICATIONS IS ILLUSTRATED BY A SHORT, BUT NONTRIVIAL EXAMPLE, FROM WHICH A NUMBER OF CONCLUSIONS ARE DRAWN REGARDING THE ART AND THE INSTRUCTION OF PROGRAMMING. (A)

THE EXAMPLE FOLLOWED THROUGH BY THIS ARTICLE ILLUSTRATES IN ITS OWN RIGHT A NUMBER OF USEFUL TECHNIQUES IN PROBLEM SOLVING, SINCE THE EXAMPLE IS A PUZZLE INVOLVING SEARCH THROUGH A LARGE SPACE. THE PROBLEM IS ONLY PROGRAMMABLE WHEN A METHOD IS FOUND THAT NARROWS THE SPACE SUFFICIENTLY TO GET ACCEPTABLE COMPUTER TIMES. (GDC)

457 SYSTEMATIC PROGRAMMING

WIRTH, N. SYSTEMATIC PROGRAMMING: AN INTRODUCTION. ENGLEWOOD CLIFFS, N. J.: PRENTICE-HALL, 1973.

DESCRIPTION:

THE PURPOSE OF THIS TEXT IS TO TEACH THE SYSTEMATIC CONSTRUCTION OF ALGORITHMS AS A PART OF A BASIC MATHEMATICAL TRAINING. IT DOES THIS PRIMARILY THROUGH PROGRAMMING EXAMPLES. THE NOTIONS OF A TOP-DOWN APPROACH ARE ESPOUSED, USING STEPWISE REFINEMENT OF THE PROGRAM DESIGN. THE ORIENTATION OF THE BOOK IS NUMERICAL COMPUTATIONS, RATHER THAN LEXICAL, PROCESS CONTROL, COMMERCIAL TRANSACTION PROCESSING, AND SO FORTH. THE DEFINITION OF PASCAL IS GIVEN AND EXAMPLES IN THE BOOK USE BOTH PASCAL STATEMENTS AND FLOWCHARTS. (GDC)

458 STRUCTURED PROGRAMMING
WIRTH, N. ON THE COMPOSITION OF WELL STRUCTURED PROGRAMS. COMPUTING SURVEYS, 1974, 6, 247-259.
DESCRIPTION:
A PROFESSIONAL PROGRAMMER'S KNOW-HOW USED TO CONSIST OF THE MASTERY OF A SET OF TECHNIQUES APPLICABLE TO SPECIFIC PROBLEMS AND TO SOME SPECIFIC COMPUTER. WITH THE INCREASE OF COMPUTER POWER, THE PROGRAMMERS' TASKS GREW MORE COMPLEX, AND HENCE, THE NEED FOR A SYSTEMATIC APPROACH BECAME EVIDENT. RECENTLY, THE SUBJECT OF PROGRAMMING METHODS, GENERALLY APPLICABLE RULES AND PATTERNS OF DEVELOPMENT, RECEIVED CONSIDERABLE ATTENTION. "STRUCTURED PROGRAMMING" IS THE FORMULATION OF PROGRAMS AS HIERARCHICAL, NESTED STRUCTURES OF STATEMENTS AND OBJECTS OF COMPUTATION. WE GIVE BRIEF EXAMPLES OF STRUCTURED PROGRAMS, SHOW THE ESSENCE OF THIS APPROACH, DISCUSS ITS RELATIONSHIP WITH PROGRAM VERIFICATION, AND COMMENT ON THE ROLE OF STRUCTURED LANGUAGES. (A)
13P, 13R.

459 PROGRAMMING LANGUAGES
WIRTH, N. AN ASSESSMENT OF THE PROGRAMMING LANGUAGE PASCAL. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 192-198.

460 SOFTWARE ENGINEERING
WITT, J. THE COLUMBUS APPROACH. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 358-363.

461 PROGRAMMER APTITUDE TEST
WOLFE, J.M. AN INTERIM VALIDATION REPORT ON THE WOLFE PROGRAMMING APTITUDE TEST (EXPERIMENTAL FORM S). COMPUTER PERSONNEL, 1977, 6(1-2), 9-11.

462 COMPUTER PERSONNEL TURNOVER
WOLFE, J.M. PERSONNEL TURNOVER RATES. COMPUTER PERSONNEL, 1977, 7(3), 6.

463 SYSTEMS ANALYSIS APTITUDE TEST
WOLFE, J.M. A VALIDATION REPORT ON THE WOLFE SYSTEMS ANALYSIS APTITUDE TEST (EXPERIMENTAL FORM B3). COMPUTER PERSONNEL, 1977, 6(1-2), 11-12.

464 SOFTWARE COSTS
WOLVERTON, R.W. THE COST OF DEVELOPING LARGE-SCALE SOFTWARE. IEEE TRANSACTIONS ON COMPUTERS, 1974, C-23, 615-636.
DESCRIPTION:
THE WORK OF SOFTWARE COST FORECASTING FALLS INTO TWO PARTS. FIRST, WE MAKE WHAT WE CALL STRUCTURAL FORECASTS, AND THEN WE CALCULATE THE ABSOLUTE DOLLAR-VOLUME FORECASTS. STRUCTURAL FORECASTS DESCRIBE THE TECHNOLOGY AND FUNCTION OF A SOFTWARE PROJECT, BUT NOT ITS SIZE. WE ALLOCATE RESOURCES (COSTS) OVER THE PROJECT'S LIFE CYCLE FROM THE STRUCTURAL FORECASTS. JUDGMENT, TECHNICAL KNOWLEDGE, AND ECONOMETRIC RESEARCH SHOULD COMBINE IN MAKING THE STRUCTURAL FORECASTS. A METHODOLOGY BASED ON A 25 X 7 STRUCTURAL FORECAST MATRIX THAT HAS BEEN USED BY TRW WITH GOOD RESULTS OVER THE PAST FEW YEARS IS PRESENTED IN THIS PAPER. WITH THE STRUCTURAL FORECAST IN HAND, WE GO ON TO CALCULATE THE ABSOLUTE DOLLAR-VOLUME FORECASTS. THE GENERAL LOGIC FOLLOWED IN "ABSOLUTE" COST ESTIMATING CAN BE BASED ON EITHER A MENTAL PROCESS OR AN EXPLICIT ALGORITHM. A COST ESTIMATING ALGORITHM IS PRESENTED AND FIVE TRADITIONAL METHODS OF SOFTWARE COST FORECASTING ARE DESCRIBED: TOP-DOWN ESTIMATING, SIMILARITIES AND DIFFERENCES ESTIMATING, STANDARDS ESTIMATING, AND BOTTOM-UP ESTIMATING. ALL FORECASTING METHODS SUFFER FROM THE NEED FOR A VALID COST DATA BASE FOR MANY ESTIMATING SITUATIONS. SOFTWARE INFORMATION ELEMENTS THAT EXPERIENCE HAS SHOWN TO BE USEFUL IN ESTABLISHING SUCH A DATA BASE ARE GIVEN IN THE BODY OF THE PAPER. MAJOR PRICING PITFALLS ARE IDENTIFIED. TWO CASE STUDIES ARE PRESENTED THAT ILLUSTRATE THE SOFTWARE COST FORECASTING METHODOLOGY AND HISTORICAL RESULTS. TOPICS FOR FURTHER WORK AND STUDY ARE SUGGESTED. (A)

465 PROGRAMMING
WOODGER, M. ON SEMANTIC LEVELS IN PROGRAMMING. INFORMATION PROCESSING 71, NORTH HOLLAND, AMSTERDAM, 1972.

466 NEED FOR ACHIEVEMENT AMONG COMPUTER PERSONNEL
WOODRUFF, C.K. THE NEED FOR ACHIEVEMENT AMONG DATA PROCESSING PERSONNEL: AN EMPIRICAL STUDY. COMPUTER PERSONNEL, 1978, 7(4).

467 COMPUTER PERSONNEL JOB SATISFACTION
WOODRUFF, C.K. JOB SATISFACYION OF DATA PROCESSING PERSONNEL. COMPUTER PERSONNEL, 1978, 7(4).

468 PROGRAMMING
WULF, W. PROGRAMMING METHODOLOGY, REPORT OF WORKSHOP 3, PROCEEDINGS OF A SYMPOSIUM ON THE HIGH COST OF SOFTWARE, MONTEREY, CALIFORNIA, SEPTEMBER 1973.

469 FAULT TOLERANCE
WULF, W.A. RELIABLE HARDWARE/SOFTWARE ARCHITECTURE. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1975, SE-1, 233-240.

470 PROGRAM VERIFICATION

YELOWITZ, L., & DUNCAN, A.G. ABSTRACTIONS, INSTANTIATIONS, AND PROOFS OF MARKING ALGORITHMS. IN PROCEEDINGS OF THE ACM SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND PROGRAMMING LANGUAGES, SIGPLAN NOTICES, AUGUST 1977, 12(8), 13-21 (ALSO SIGART NEWSLETTER, AUGUST 1977, NO. 64, 13-21).

DESCRIPTION:

A DETAILED LOOK IS TAKEN AT THE PROBLEM OF FACTORING PROGRAM PROOFS INTO A PROOF OF THE UNDERLYING ALGORITHM, FOLLOWED BY A PROOF OF CORRECT IMPLEMENTATION OF ABSTRACT VARIABLES AT THE CONCRETE LEVEL. WE DO THIS CONSIDERING FOUR DIFFERENT CONCRETE 'MARKING' ALGORITHMS AND FORMULATING A SINGLE ABSTRACT ALGORITHM AND SET OF ABSTRACT SPECIFICATIONS THAT CAN BE INSTANTIATED TO EACH OF THE FOUR CONCRETE CASES. AN INTERMEDIATE ASSERTION, AS WELL AS SUFFICIENT CONDITIONS FOR CORRECT INITIALIZATION, INVARIANCE, AND CORRECTNESS AT TERMINATION ARE GIVEN AT THE ABSTRACT LEVEL. PROOFS AT THE CONCRETE LEVEL ARE THEN GIVEN BY EXHIBITING APPROPRIATE MAPPING FUNCTIONS (FROM THE CONCRETE STATE VECTOR TO THE ABSTRACT VARIABLES), AND SHOWING THAT THE SUFFICIENT CONDITIONS ARE TRUE. PROOFS OF TERMINATION ARE GIVEN BY INSTANTIATING 'TERMINATION SCHEMAS'. (A)
9P, 13R.

471 GENERAL

YNTEMA, D.B. THE SOFTWARE PROBLEM (TECHNICAL REPORT NO. 1964-5). CAMBRIDGE, MASSACHUSETTS: MASSACHUSETTS INSTITUTE OF TECHNOLOGY, LINCOLN LABORATORY, SEPTEMBER 1964.

472 PROGRAMMING PRACTICES

YOHE, J.M. AN OVERVIEW OF PROGRAMMING PRACTICES. COMPUTING SURVEYS, 1974, 6, 221-245.

DESCRIPTION:

THE PURPOSE OF THIS PAPER IS TO INDICATE SOMETHING OF THE NATURE OF "GOOD PROGRAMMING." IN THIS CONTEXT, PROGRAMMING IS TAKEN TO MEAN THE ENTIRE PROCESS OF COMMUNICATION BETWEEN HUMANS AND COMPUTERS. THE PROGRAMMING PROCESS IS SUBDIVIDED INTO NINE TASKS, AND AN ELEMENTARY DISCUSSION OF EACH OF THESE TASKS IS PRESENTED. ALTHOUGH THE PAPER IS PRIMARILY DIRECTED TO THE STUDENT OR NOVICE PROGRAMMER, MORE EXPERIENCED PEOPLE MAY FIND IT A WORTHWHILE AID IN CODIFYING OR REINFORCING THEIR EXPERIENCE. (A)
25P, 41R.

473 PROGRAMMING ERRORS

YOUNGS, E.A. ERROR-PRONENESS IN PROGRAMMING (UNPUBLISHED DOCTORAL DISSERTATION TION, UNIVERSITY OF NORTH CAROLINA). 1970.

DESCRIPTION:

AN EXPERIMENT WAS CONDUCTED TO EXAMINE THE TYPES AND FREQUENCIES OF ERRORS MADE BY BEGINNING AND EXPERIENCED PROGRAMMERS USING COBOL, ALGOL, PL/1, FORTRAN, AND BASIC. SUGGESTIONS ARE MADE FOR IMPROVEMENTS IN PROGRAMMING LANGUAGES AND COMPILERS. PERHAPS THE MOST IMPORTANT CONTRIBUTION OF THIS PAPER IS TO SUGGEST TECHNIQUES AND CONCEPTS FOR THE QUANTITATIVE STUDY OF PROGRAM DEVELOPMENT. (MEA)
153P, 19R.

474 PROGRAMMING ERRORS
YOUNGS, E.A. HUMAN ERRORS IN PROGRAMMING. INTERNATIONAL JOURNAL OF MAN-MACHINE STUDIES, 1974, 6, 361-376.
DESCRIPTION:
THIS STUDY ATTEMPTS TO SYSTEMATIZE THE DESCRIPTION OF THE ERRORS THAT PROGRAMMERS MAKE. BY COLLECTING PROTOCOL DATA FROM 42 PROGRAMMERS, SOME INSIGHTS CONCERNING THE RELATIVE IMPORTANCE OF VARIOUS PROGRAMMING ERRORS ARE ACHIEVED. THESE INSIGHTS ARE INTERPRETED IN TERMS OF PROGRAMMER EXPERIENCE AND THE DESIGN AND REDESIGN OF GENERAL PURPOSE, COMPILER-TYPE PROGRAMMING LANGUAGES. (A)
16P, 6R.

475 SOFTWARE DESIGN
YOURDON, E., & CONSTANTINE, L.L. STRUCTURED DESIGN. NEW YORK: YOURDON, INC., 1975.

476 ERROR DETECTION
ZELKOWITZ, M.V., MCMULLEN, P.R., MERKEL, K.R., & LARSEN, H.J. ERROR CHECKING WITH POINTER VARIABLES. IN ACM '76: PROCEEDINGS OF THE ANNUAL CONFERENCE. NEW YORK, NEW YORK: ASSOCIATION FOR COMPUTING MACHINERY, 1976, 391-395.

477 SOFTWARE PHYSICS
ZISLIS, P. AN EXPERIMENT IN ALGORITHM IMPLEMENTATION (TECHNICAL REPORT NO. CSD-TR-96). LAFAYETTE, INDIANA: PURDUE UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, 1973.
DESCRIPTION:
PRESENTED VARIOUS MEASURES (E.G., SPECIFICATION TIMES, TIMING BREAKDOWNS FOR CODED PROGRAMS, AND SOFTWARE PHYSICS PARAMETERS) FOR 12 ALGORITHMS FROM CACM. DATA ISSUED IN A LETTER PAPER BY HALSTEAD AND ZISLIS (TECHNICAL REPORT CSD-TR-97). (O)
17P, 0R.

478 SOFTWARE PHYSICS
ZWEBEN, S.H. SOFTWARE PHYSICS: RESOLUTION OF AN AMBIGUITY IN THE COUNTING PROCEDURE (TECHNICAL REPORT NO. CSD-TR-93). LAFAYETTE, INDIANA: PURDUE UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE, UNDATED.
DESCRIPTION:
THIS PAPER ATTEMPTS TO SHOW WHY THE COMMA SHOULD BE TREATED AS A SEPARATE OPERATOR. (O)
7P, 4R.

AUTHOR INDEX

AUTHOR INDEX

AUTHOR INDEX

AUTHOR INDEX

AUTHOR INDEX

AUTHOR INDEX

AUTHOR INDEX

AUTHOR INDEX

AUTHOR INDEX

DOCUMENT SOURCE INDEX, PROCEEDINGS

INSTITUTION INDEX

INSTITUTION INDEX

INSTITUTION INDEX

INSTITUTION INDEX

SUBJECT INDEX

SUBJECT INDEX

SUBJECT INDEX